The Ideal Continual Learner: An Agent That Never Forgets

Liangzu Peng [1 2] **Paris V. Giampouras** [1] **René Vidal** [2 3]

Abstract

The goal of continual learning is to find a model that solves multiple learning tasks which are presented sequentially to the learner. A key challenge in this setting is that the learner may *forget* how to solve a previous task when learning a new task, a phenomenon known as *catastrophic forgetting*. To address this challenge, many practical methods have been proposed, including memory-based, regularization-based, and expansion-based methods. However, a rigorous theoretical understanding of these methods remains elusive. This paper aims to bridge this gap between theory and practice by proposing a new continual learning framework called *Ideal Continual Learner (`ICL`)*, which is guaranteed to avoid catastrophic forgetting by construction. We show that `ICL` unifies multiple well-established continual learning methods and gives new theoretical insights into the strengths and weaknesses of these methods. We also derive generalization bounds for `ICL` which allow us to theoretically quantify *how rehearsal affects generalization*. Finally, we connect `ICL` to several classic subjects and research topics of modern interest, which allows us to make historical remarks and inspire future directions.

1. Introduction

The goal of building intelligent machines that are adaptive and self-improving over time has given rise to the *continual learning* paradigm, where the *learner* needs to solve multiple tasks presented sequentially (Thrun & Mitchell, 1995). In this setting, a key challenge known as *catastrophic forgetting* (McCloskey & Cohen, 1989) is that, unlike humans, the agent may *forget* how to solve past tasks (i.e., exhibit larger errors on past tasks), after learning the present one. To address this problem, many practical methods have been proposed. For example, ***memory-based*** methods store data from past tasks and use it to solve the present task (Lopez-Paz & Ranzato, 2017), ***regularization-based*** methods add some regularization terms to the loss in order to prevent overfitting to the current task (Kirkpatrick et al., 2017), and ***expansion-based*** methods expand the network architecture to accommodate learning a new task (Rusu et al., 2016).

Deep continual learning methods that embody either of these three ideas, or combinations thereof, have greatly improved the empirical performance in resisting catastrophic forgetting (Liu, 2022).[1] While these methods have evolved so rapidly, at least two questions remain under-explored:

(Q1) *Is there any **mathematical** connection between continual learning and other conceptually related fields*? The answer seems elusive even for *multitask learning* (i.e., joint training of all tasks): Chaudhry et al. (2020) and Mirzadeh et al. (2021), among many others, advocated that performing multitask learning gives the best performance for continual learning, while Wu et al. (2022) claimed the opposite.

(Q2) *How does the rehearsal mechanism **provably** affect generalization*? Lopez-Paz & Ranzato (2017) claimed that *rehearsal* (i.e., joint training with part of previous data stored in memory and the data of the present task) would result in overfitting (and jeopardize generalization). Chaudhry et al. (2019) rebutted against this claim by empirical evidence, while Verwimp et al. (2021) offered empirical counter-evidence for what Chaudhry et al. (2019) argued. Indeed, the answer to (Q2) has been unclear, which is why Verwimp et al. (2021) posed this as a serious open problem.

Our Contributions. In this paper, we focus on theoretically understanding continual learning and catastrophic forgetting by trying to answer Questions (Q1) and (Q2). In particular:

- We propose a general framework for continual learning, called the *Ideal Continual Learner (`ICL`)*, and we show that, under mild assumptions, `ICL` *never forgets*. This characterization of never forgetting makes it possible to address Questions (Q1) and (Q2) via dissecting the optimization and generalization properties of `ICL`.

[1]Mathematical Institute for Data Science, Johns Hopkins University, Baltimore, USA [2]Innovation in Data Engineering and Science (IDEAS), University of Pennsylvania, Philadelphia, USA [3]NORCE Norwegian Research Centre, Norway. Correspondence to: Liangzu Peng <lpenn@seas.upenn.edu>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

[1]We discuss the most relevant references in the main paper as we proceed, and we review related works in detail in the appendix.

§2		Optimization Basics of Continual Learning
	§2.2 & §2.3	Definition of ICL (Definition 1) & ICL never forgets (Proposition 1)
	§2.4 & §2.5	ICL is a bilevel optimizer (Propositions 2 and 3) & ICL is a multitask learner (Proposition 4)
§3		Examples of ICL & Understanding Deep Continual Learning
	§3.1.3	ICL is a ***memory-based (projection-based) method*** (Proposition 6 and Remark 3)
	§3.2.3	ICL is an ***expansion-based method*** & wide neural networks forget less catastrophically
§4		Generalization Basics of Continual Learning
	§4.3	ICL and constrained learning & ICL is a ***regularization-based method***
	§4.4	Generalization guarantees of rehearsal (Theorem 3) & remarks on memory selection methods

Table 1: Structure and messages of the paper.

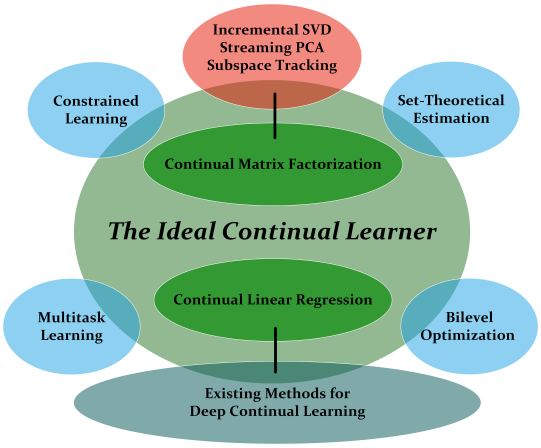


Figure 1: The *Ideal Continual Learner* (ICL) and related subjects. The exact connections between ICL and these subjects are discussed throughout the paper and appendix.

- Question (Q1) is considered throughout the paper: We bring to light the connection of ICL to many other fields—visualized in Figure 1—including *set-theoretical estimation* (§2.3), *bilevel optimization* (§2.4), multitask learning (§2.5), deep continual learning (§3), *incremental SVD* (§3.2), and *constrained learning* (§4.2). In particular, we dissect ICL in two examples, *continual learning regression* (§3.1) and *continual matrix factorization* (§3.2), showing that ICL is a ***memory-based optimization method*** for the former example and an ***expansion-based*** for the latter. We also connect ICL to ***regularization-based*** methods (§4.2). These shed considerable lights on many aspects of existing deep continual learning methods, e.g., their failure cases, the role of memory and network widths, and so on. With all these connections, we provide historical context and novel insights for future research avenues.
- Question (Q2) is explored in §4, where we prove the generalization properties of ICL based on the classic statistical learning frameworks. Crucially, our theory quantifies how rehearsal influences generalization, shedding light on (Q2).

Since the paper consists of multiple messages relevant to continual learning, it might be beneficial to overview them in Table 1, which includes direct links to the contents and aims at helping the reader navigate.

2. Continual Learning Basics: Optimization

2.1. Problem Setup

Consider tasks $1, \ldots, T$, where task t can be solved by minimizing some problem of the form

$$\mathcal{G}_t := \operatorname*{argmin}_{\boldsymbol{w} \in \mathcal{W}} L_t(\boldsymbol{w}; D_t). \tag{1}$$

Here, D_t is the given data set for task t, and $\mathcal{G}_t$ is the set of global minimizers of the objective L_t (1). By *continual learning*, we mean solving the tasks sequentially from task 1 to task T and finding some ground truth in the *hypothesis space* $\mathcal{W} \subset \mathbb{R}^n$ that minimizes all losses (1); if an algorithm can do so, then we say it never *forgets*. While each task could have a different hypothesis space, we trade this generality for simplifying the presentation.

For such ground truth to exist, we need a basic assumption:

Assumption 1 (Shared Multitask Model†)**.** All tasks (1) share a common global minimizer, i.e., $\cap_{t=1}^{T} \mathcal{G}_t \neq \varnothing$.

The intuition behind Assumption 1, which generalizes those of Evron et al. (2022) and Peng & Risteski (2022), is that if $\cap_{t=1}^{T} \mathcal{G}_t = \varnothing$ then there is no shared global minimizer and continual learning without forgetting is infeasible. Assumption 1 might be relaxed into the existence of *approximate* common global minimizers; we do not pursue this idea here.

Note that verifying whether $\cap_{t=1}^{T} \mathcal{G}_t$ is empty or not is in general NP-hard even when every $\mathcal{G}_t$ is a "simple" polytope (Tiwary, 2008), which is the main reason that Knoblauch

et al. (2020) asserts continual learning without forgetting is in general NP-hard. However, Assumption 1 sidesteps the curse of the NP-hardness, and makes it possible to solve the continual learning problem computationally efficiently.

2.2. The Ideal Continual Learner† ($\texttt{ICL}^\dagger$)

The main role of this section and the paper, is this:

Definition 1 (The Ideal Continual Learner†, $\texttt{ICL}^\dagger$)**.** With $\mathcal{K}_0 := \mathcal{W}$, $\texttt{ICL}^\dagger$ is an algorithm that solves the following program sequentially for $t = 1, 2, \ldots, T$:

$$\mathcal{K}_t \leftarrow \operatorname*{argmin}_{\boldsymbol{w} \in \mathcal{K}_{t-1}} L_t(\boldsymbol{w}; D_t). \tag{2}$$

The symbol $\dagger$ in Definition 1 is a reminder that $\texttt{ICL}^\dagger$ has not been proved (until §4) to be a learner in the *statistical learning* sense. Recursion (2) asks for a lot: Computing the entire set $\mathcal{K}_1$ of global minimizers is hard enough, let alone constraining over it, recursively! Indeed, we use the word "*ideal*" to imply that $\texttt{ICL}^\dagger$ is *not realizable* at the current stage of research. However, take a leap of faith, and we will see many pleasant consequences of $\texttt{ICL}^\dagger$ under Assumption 1, where the *theoretical significance* of $\texttt{ICL}^\dagger$ is treasured.

An immediate observation is that $\mathcal{W} = \mathcal{K}_0 \supset \cdots \supset \mathcal{K}_T$, that is $\mathcal{K}_t$ *shrinks* (more precisely, does not *grow*) over time. An analogy can be made from two perspectives. From a *human learning* perspective, $\mathcal{K}_t$ can be viewed as a *knowledge representation*: After continually reading a 1000-page book, one internalizes the knowledge and represents the whole book with a few key notations or formulas, or with a single cheatsheet (but see also Remark 5, *Grow* = *Shrink*). From a *control* perspective, the analogy is that the *uncertainty* $\mathcal{K}_t$ over the true solutions reduces as $\texttt{ICL}^\dagger$ learns from data and tasks. In fact, as we will review in Appendix E, $\texttt{ICL}^\dagger$ is closely related to *set-theoretical estimation* in control (Combettes, 1993; Kieffer et al., 1998).

2.3. $\texttt{ICL}^\dagger$ is Sufficient and Minimal

In this section, we show that the Ideal Continual Learner† is *sufficient* (Fisher, 1922) and *minimal* (Lehmann & Scheffé, 1950), which are two general and fundamental properties in the information-theoretical and statistical sense.

Sufficiency. It has been held that "... *preventing forgetting by design is therefore not possible [even if all losses* L_t (1) *are the same]*" (van de Ven et al., 2022). Nevertheless, under Assumption 1, $\texttt{ICL}^\dagger$ never forgets by design:

Proposition 1 (Sufficiency)**.** *Under Assumption 1,* $\texttt{ICL}^\dagger$ *solves all tasks optimally. In other words, we have* $\mathcal{K}_t = \cap_{i=1}^t \mathcal{G}_i$ *for every* $t = 1, \ldots, T$.

Minimality. Here we show (with rigor) that the knowledge representation $\mathcal{K}_t$ is *minimal* (given the order of the tasks). Consider the first two tasks, which are associated with objective functions L_1 and L_2 and global minimizers $\mathcal{G}_1$ and $\mathcal{G}_2$ respectively. Assume that $\mathcal{G}_1$ and $\mathcal{G}_2$ intersect (Assumption 1), and let $\hat{\boldsymbol{w}} \in \mathcal{G}_1 \cap \mathcal{G}_2$. A learner solves the first task by minimizing L_1 (1), stores some information $\mathcal{I}_1$, and proceeds to the second (the stored information $\mathcal{I}_1$ could consist of some data samples, gradients, global minimizers, and so on). If the stored information $\mathcal{I}_1$ is not enough to reveal that $\hat{\boldsymbol{w}}$ is optimal to task 1, then it is impossible for the learner to figure out that $\hat{\boldsymbol{w}}$ is actually simultaneously optimal to both tasks—even if it could find later that $\hat{\boldsymbol{w}}$ is optimal to task 2. Thus, either the learner would conclude that no common global minimizer exists and Assumption 1 is violated, or catastrophic forgetting inevitably takes place. In an independent effort, Peng & Risteski (2022) made a similar argument specifically for two-layer neural networks. Our argument is more general, and is for a different purpose.

Note that $\hat{\boldsymbol{w}}$ can be any element of $\mathcal{K}_2 = \mathcal{G}_1 \cap \mathcal{G}_2$. Thus, storing a proper subset of $\mathcal{K}_2$ is sub-optimal since that might exclude global minimizers (e.g., $\hat{\boldsymbol{w}}$) of subsequent tasks corresponding to $t > 2$ and lead to catastrophic forgetting. In light of this, we observe that catastrophic forgetting can only be prevented if we store the entire set $\mathcal{K}_t$ or its *equivalent information* (we will soon see some equivalent representations of $\mathcal{K}_t$). In this sense, we say the knowledge representation $\mathcal{K}_t$ is minimal. It is the two properties, sufficiency and minimality, which $\texttt{ICL}^\dagger$ (Definition 1) enjoys, that justify the naming: *the* Ideal Continual Learner† that never forgets.

2.4. $\texttt{ICL}^\dagger$ = Bilevel Optimizer

Since the constraint $\boldsymbol{w} \in \mathcal{K}_{t-1}$ of (2) requires $\boldsymbol{w}$ to be a global minimizer of all previous tasks, for each task $\texttt{ICL}^\dagger$ needs to solve a bilevel program (2), which is in general difficult (Vicente & Calamai, 1994; Jiang et al., 2022). However, with Assumption 1, we can describe (2) in a relatively simple way. The first description is immediate:

Proposition 2. *Under Assumption 1, for every* $t = 1, \ldots, T$, *the recursion* (2) *of* $\texttt{ICL}^\dagger$ *is equivalent to*

$$\begin{aligned} &\min_{\boldsymbol{w} \in \mathcal{W}} L_t(\boldsymbol{w}; D_t) \\ &\text{s.t. } L_i(\boldsymbol{w}; D_i) \le c_i, \ \forall i = 1, \ldots, t-1, \end{aligned} \tag{3}$$

where c_i *is the minimum value of* $L_i(\boldsymbol{w}; D_i)$ *over* $\mathcal{W}$, *computed during solving previous tasks.*

In Proposition 2, the inequality of $L_i(\boldsymbol{w}; D_i) \le c_i$ can be replaced by equality $=$, and the constraint of (3) is understood as trivially fulfilled if $t = 1$.

Remark 1 ($\mathcal{K}_t$ = Loss + Data). If we store all data and losses, we can recover $\mathcal{K}_t$ via minimizing (3), hence (obviously) memorizing data and losses can prevent forgetting.

A formulation similar to (3) is known in the literature; see, e.g., Lopez-Paz & Ranzato (2017). However, as in many

continual learning papers, their formulation is dominated by computational considerations in the deep learning context, e.g., their c_i is the loss of a sufficiently trained deep network for the i-th task over part of data samples; it is not necessarily a minimum value. Such approach, though practically important, makes it hard to derive theoretical properties.

$\texttt{ICL}^\dagger$ is also equivalent to the following formulation:

Proposition 3. *Assume objective functions $L_1, \ldots, L_T$ are convex and differentiable. Let $\mathcal{W} = \mathbb{R}^n$. Under Assumption 1, each step (2) of $\texttt{ICL}^\dagger$ is equivalent to*

$$\begin{aligned} &\min_{\boldsymbol{w} \in \mathbb{R}^n} L_t(\boldsymbol{w}; D_t) \\ &\text{s.t. } \nabla L_i(\boldsymbol{w}; D_i) = 0, \ \forall i = 1, \ldots, t-1, \end{aligned} \tag{4}$$

Remark 2 ($\mathcal{K}_t =$ Gradient Equations). If we store all equations $\nabla L_i(\boldsymbol{w}; D_i) = 0$, we can recover $\mathcal{K}_t$ via solving (4), so memorizing these equations resists forgetting.

More generally, a common idea in bilevel programming is to rewrite the *lower-level problem* (e.g., $\boldsymbol{w} \in \mathcal{K}_{t-1}$) as KKT conditions, and many algorithms exist for the latter formulation; see, e.g., Dempe & Dutta (2012) and the follow-up works. While it is beyond the scope of the paper, contextualizing these ideas for implementing $\texttt{ICL}^\dagger$ (2) is an important direction that would inspire novel and theoretically grounded continual learning algorithms. Note that bilevel programming has been used for continual learning (Borsos et al., 2020), but the use is mainly for selecting which data samples to store in the memory.

2.5. $\texttt{ICL}^\dagger$ = Multitask Learner†

Under Assumption 1, the following connection between continual learning and multitask learning arises naturally:

Proposition 4 ($\texttt{ICL}^\dagger$ = Multitask Learner†)**.** *Let $\alpha_1, \ldots, \alpha_t$ be arbitrary positive numbers. Recall $\mathcal{K}_t$ is defined in (2) as the output of $\texttt{ICL}^\dagger$. Under Assumption 1, we have*

$$\mathcal{K}_t = \operatorname*{argmin}_{w \in \mathcal{W}} \sum_{i=1}^{t} \alpha_i L_i(\boldsymbol{w}; D_i), \quad \forall t = 1, \ldots, T. \tag{5}$$

Assumption 1 implies that $\texttt{ICL}^\dagger$ never forgets and provides the best performance for continual learning, and that minimizing the multitask loss (5) recovers $\texttt{ICL}^\dagger$. Hence, under Assumption 1, minimizing the multitask loss yields the best performance for continual learning. This simple result addresses issues pertaining to Question (Q1).

3. Examples of $\texttt{ICL}^\dagger$

This section discusses $\texttt{ICL}^\dagger$ for two examples, *continual linear regression* (§3.1), *continual matrix factorization* (§3.2). By way of examples, we acquire some understanding of existing deep continual learning practices (§3.1.3, §3.2.3).

3.1. Continual Linear Regression

In §3.1.1, we review the problem setup of continual linear regression (Evron et al., 2022). In §3.1.2 we present the implementation of $\texttt{ICL}^\dagger$ for continual linear regression. In §3.1.3 we draw connections to deep continual learning, shedding light on ***memory-based optimization methods***.

3.1.1. Background and Problem Setup

In *continual* linear regression, the data $D_t := (\boldsymbol{X}_t, \boldsymbol{y}_t)$ of each task t consists of a feature matrix $\boldsymbol{X}_t \in \mathbb{R}^{m_t \times n}$ and a response vector $\boldsymbol{y}_t \in \mathbb{R}^{m_t}$; here m_t is the number of samples of task t and n their dimension, and the loss L_t is

$$L_t(\boldsymbol{w}; (\boldsymbol{X}_t, \boldsymbol{y}_t)) = \|\boldsymbol{X}_t \boldsymbol{w} - \boldsymbol{y}_t\|_2^2. \tag{6}$$

The set $\mathcal{G}_t$ of global minimizers for each task (6) is exactly an affine subspace. Under Assumption 1, the intersection of these affine subspaces is not empty, and finding one element in that intersection would solve all tasks optimally. Note that, if $\boldsymbol{X}_t$ is of full column rank, then solving task t yields a unique solution, which is optimal to all tasks by Assumption 1. To avoid this trivial case, we follow Evron et al. (2022) and assume $\boldsymbol{X}_t$ is rank-deficient for every t. Studying continual regression is of interest as (continual) regression is closely related to deep (continual) learning in the *neural tangent kernel* regime (Jacot et al., 2018). Moreover, as will be shown in §3.1.3, $\texttt{ICL}^\dagger$ connects continual linear regression to deep continual learning even more tightly.

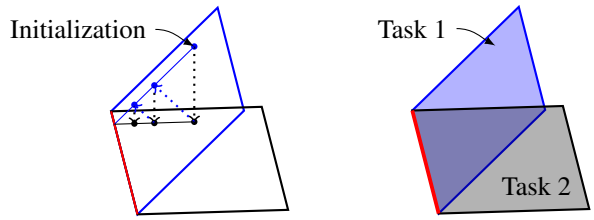
With some initialization, the algorithm of Evron et al. (2022) proceeds in a successive (or alternating) projection fashion (Figure 2a). When presented with task t, it projects onto the affine subspace $\mathcal{G}_t = \{\boldsymbol{w} \in \mathbb{R}^n : \boldsymbol{X}_t^\top \boldsymbol{X}_t \boldsymbol{w} = \boldsymbol{X}_t^\top \boldsymbol{y}_t\}$; this projection can be implemented via invoking a singular value decomposition (SVD) or leveraging the implicit bias of (stochastic) gradient descent. Evron et al. (2022) employed the latter approach as it is *memoryless*, i.e., it does not use any extra storage except the estimate and current data. However, being memoryless might entail catastrophic forgetting in the worst case (Evron et al., 2022, Theorem 7).

3.1.2. Implementing $\texttt{ICL}^\dagger$

Our $\texttt{ICL}^\dagger$ method is illustrated in Figure 2b. As per (2), $\texttt{ICL}^\dagger$ sets $\mathcal{W} \leftarrow \mathbb{R}^n$, finds the affine subspace $\mathcal{G}_1$ of task 1, uses it to regularize task 2, finds their common solutions $\mathcal{G}_1 \cap \mathcal{G}_2$, and so forth. We describe some more details next.

Let $\hat{\boldsymbol{w}}_t \in \mathcal{K}_t$ be a common global minimizer to tasks $1, \ldots, t$. Let $\boldsymbol{K}_t$ be an orthonormal basis matrix[2] for the intersection of the nullspaces of $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_t$. Note that

[2] To simplify the presentation, we did not annotate some matrix sizes or subspace dimensions. It is understood that such matrices are of suitable sizes and subspaces of appropriate dimensions.



(a) Evron et al. (2022) (b) $\texttt{ICL}^\dagger$ (Ours)

Figure 2: For two tasks, Evron et al. (2022) follow the arrows (2a) and project the current estimate onto the affine subspace of solutions (blue or black), in an alternating fashion, eventually reaching a common solution (red). Instead, $\texttt{ICL}^\dagger$ solves task 1, stores the affine subspace (2b, blue), uses it to regularize task 2 and finds all solutions (2b, red).

$\boldsymbol{w} \in \mathcal{K}_t$ if and only if $\boldsymbol{w}$ can be written as $\boldsymbol{w} = \hat{\boldsymbol{w}}_t + \boldsymbol{K}_t \boldsymbol{a}$ for some coefficient vector $\boldsymbol{a}$, so for implementing $\texttt{ICL}^\dagger$ we will compute and store $(\hat{\boldsymbol{w}}_t, \boldsymbol{K}_t)$. Such computation is summarized in the following proposition:

Proposition 5. *$\texttt{ICL}^\dagger$ for continual linear regression can be implemented as follows. Given data $(\boldsymbol{X}_1, \boldsymbol{y}_1)$, one can compute $(\hat{\boldsymbol{w}}_1, \boldsymbol{K}_1)$ via SVD. With $t > 1$, given $(\hat{\boldsymbol{w}}_{t-1}, \boldsymbol{K}_{t-1})$ and data $(\boldsymbol{X}_t, \boldsymbol{y}_t)$, one can compute $(\hat{\boldsymbol{w}}_t, \boldsymbol{K}_t)$ via solving*

$$\min_{\boldsymbol{w} \in \mathcal{K}_{t-1}} \|\boldsymbol{X}_t \boldsymbol{w} - \boldsymbol{y}_t\|_2^2 \tag{7}$$

$$\Leftrightarrow \min_{\boldsymbol{a}} \|\boldsymbol{X}_t(\hat{\boldsymbol{w}}_{t-1} + \boldsymbol{K}_{t-1}\boldsymbol{a}) - \boldsymbol{y}_t\|_2^2 \tag{8}$$

In particular, (8) can be solved via SVD.[3]

Compared to the method of Evron et al. (2022), $\texttt{ICL}^\dagger$ uses some extra storage for $\boldsymbol{K}_t$ to attain optimality and resist catastrophic forgetting; note though that the storage consumption never grows as $\mathcal{K}_T \subset \cdots \subset \mathcal{K}_1$. Finally, note that for continual linear regression, one can formulate $\texttt{ICL}^\dagger$ in different ways than (8); see, e.g., Proposition 5.5 of (Evron et al., 2023) (based on Fisher information) and Proposition 3 (based on the first-order optimality conditions).

3.1.3. Connections to Deep Continual Learning

We first connect $\texttt{ICL}^\dagger$ to the *orthogonal gradient descent* method (OGD) of Farajtabar et al. (2020):

Proposition 6 (Informal)**.** *When applied to continual linear regression, the OGD method consists of the updates*

$$\boldsymbol{w}^+ \leftarrow \boldsymbol{w} - \gamma \boldsymbol{K}_{t-1}\boldsymbol{K}_{t-1}^\top \boldsymbol{h}, \tag{9}$$

where $\boldsymbol{h}$ is the gradient of the objective of (7), γ stepsize, and $\boldsymbol{w}$ (resp. $\boldsymbol{w}^+$) the previous (resp. current) iterate. Moreover, OGD converges to a global minimizer of (7).

We prove Proposition 6 in Appendix C, where we also review the OGD method; see Bennani et al. (2020); Doan et al. (2021) for different theoretical aspects of OGD.

Since regression can be viewed as a single-layer linear network with a least-squares loss, formula (9) can be extended into deep continual learning in a *layer-wise* manner. This viewpoint allows us to make the following connection:

Remark 3. For continual linear regression, the methods of Zeng et al. (2019); Saha et al. (2021); Wang et al. (2021b); Kong et al. (2022) are of the form (9), and so they all converge "approximately" to global minimizers of (7); these methods differ mainly in how the projection $\boldsymbol{K}_{t-1}\boldsymbol{K}_{t-1}^\top$ is approximated and stored. For deep continual learning, their methods perform (9) in a *layer-wise* manner, with different projections to update the parameters of every linear layer.

We review the methods of Remark 3 in Appendix D.1. By showing that these methods can be derived from the principled objective (7) of $\texttt{ICL}^\dagger$, we make them more interpretable. For example, Kong et al. (2022) approximate $\boldsymbol{K}_{t-1}\boldsymbol{K}_{t-1}^\top$ better than Zeng et al. (2019); not surprisingly, Kong et al. (2022) gets better performance (see their Table 1). Moreover, we provide an important failure case. Note that all these methods (including OGD) approximately solve (7) and the correctness of (7) relies on Assumption 1. As a consequence, in the absence of Assumption 1, all these methods (and $\texttt{ICL}^\dagger$) might be *obsessed with the past*:

Example 1 (Past = Present)**.** Consider the continual linear regression problem (6), and assume $\mathcal{G}_1$ and $\mathcal{G}_2$ are two parallel and non-intersecting affine subspaces. In this situation, $\texttt{ICL}^\dagger$ will compute $\mathcal{K}_1 = \mathcal{G}_1$ and $\mathcal{K}_2 = \mathcal{G}_1$. In other words, it will get stuck to optimal points of task 1, contained in $\mathcal{K}_1$, failing to track optimal points of task 2.

3.2. Continual Matrix Factorization

The structure of this section parallels that of §3.1. In §3.2.1. We introduce the problem of *continual matrix factorization*; this problem arises as a generalization of Peng & Risteski (2022). In §3.2.2, we present the implementation of $\texttt{ICL}^\dagger$ for this problem. In §3.2.3, we make connections to deep continual learning, highlighting ***expansion-based methods***.

3.2.1. Background and Problem Setup

The *continual* matrix factorization setting is as follows. The data $D_t := \boldsymbol{Y}_t$ of task t is a matrix $\boldsymbol{Y}_t$, and every column of

[3] The use of SVD is to find singular vectors corresponding to the extreme singular values (in particular, zero singular values) of a given matrix. A basic fact from numerical linear algebra is that, in general, the extreme singular vectors of a given matrix can only be found iteratively and therefore inexactly (Trefethen & Bau, 1997, Lecture 25). Thus, the implementations that we suggested for continual linear regression (§3.1.2) and continual matrix factorization (§3.2.2) only *approximately* implement $\texttt{ICL}^\dagger$; note though that such approximation quality is typically very high due to the existence of industry-strength SVD algorithms.

$\boldsymbol{Y}_t$ lies in some (linear) subspace $\mathcal{S}_t$ of $\mathbb{R}^n$; we assume the columns of $\boldsymbol{Y}_t$ span $\mathcal{S}_t$ without loss of generality. Each task t consists of factorizing $\boldsymbol{Y}_t$ into two matrices $\boldsymbol{U}$ and $\boldsymbol{C}$ such that $\boldsymbol{U}\boldsymbol{C} = \boldsymbol{Y}_t$. This corresponds to minimizing

$$L_t\big((\boldsymbol{U}, \boldsymbol{C}); \boldsymbol{Y}_t\big) = \|\boldsymbol{U}\boldsymbol{C} - \boldsymbol{Y}_t\|_{\mathrm{F}}^2, \tag{10}$$

which is a *matrix factorization* problem. With the identity matrix I,[2], we assume $\boldsymbol{U}^\top \boldsymbol{U} = I$. The goal of *continual* matrix factorization is to factorize the whole data matrix $[\boldsymbol{Y}_1 \ \cdots \ \boldsymbol{Y}_T]$ into $\boldsymbol{U}$ and $\boldsymbol{C}$ such that $[\boldsymbol{Y}_1 \ \cdots \ \boldsymbol{Y}_T] = \boldsymbol{U}\boldsymbol{C}$, under the constraint that $\boldsymbol{Y}_t$ is presented sequentially. Under certain conditions, matrix factorization (10) is equivalent to two-layer linear neural networks, with $\boldsymbol{C}$ being the first layer of weight parameters and $\boldsymbol{U}$ the second (Baldi & Hornik, 1989; Vidal, 2020; Peng & Risteski, 2022). In that sense, analyzing continual matrix factorization would facilitate understanding deep continual learning.

Peng & Risteski (2022) performed one such analysis, based on prior works on orthogonal gradient descent (Farajtabar et al., 2020; Chaudhry et al., 2020) and matrix factorization (Ye & Du, 2021). They assumed that the rank r of $[\boldsymbol{Y}_1 \ \cdots \ \boldsymbol{Y}_T]$ is known, and that each $\boldsymbol{Y}_t$ is a vector. With r given, Peng & Risteski (2022) maintain a basis matrix $\boldsymbol{U}$ with r columns throughout the learning process. Their method is not memory-efficient for two reasons: (i) Storing r columns is unnecessary when the learner has not encountered the r-th sample (this extra consumption of memory is negligible for small r, though); (ii) their algorithm furthermore requires storing two projection matrices. In §3.2.2, we will show that $\texttt{ICL}^\dagger$ can handle the more general situation where the rank r is unknown, and it also overcomes the memory inefficiency of Peng & Risteski (2022).

3.2.2. Implementing $\texttt{ICL}^\dagger$

To fully understand $\texttt{ICL}^\dagger$ for continual matrix factorization, we first establish a basic geometric understanding of the problem and then discuss how to store $\mathcal{K}_t$ efficiently, and finally, we describe the implementation details.

Basic Subspace Geometry. Minimizing (10) in variable $\boldsymbol{C}$ with $\boldsymbol{U}$ fixed reveals that the optimal $\boldsymbol{C}$ is exactly $\boldsymbol{U}^\top \boldsymbol{Y}_t$, so (10) is equivalent to $\|\boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{Y}_t - \boldsymbol{Y}_t\|_{\mathrm{F}}^2$, an objective function for *principal component analysis* (PCA). In this way, continual matrix factorization relates to *streaming PCA* (Mitliagkas et al., 2013; Peng & Risteski, 2022), *incremental SVD* (Bunch & Nielsen, 1978), and *subspace tracking* (Balzano et al., 2018). We review these subjects in Appendix F to highlight the connection to $\texttt{ICL}^\dagger$.

Geometrically, every orthonormal basis of any subspace[2] containing $\mathcal{S}_t$ is a global minimizer of this PCA objective. In other words, any global minimizer of (10) is of the form $(\boldsymbol{U}, \boldsymbol{U}^\top \boldsymbol{Y}_t)$, where $\boldsymbol{U}$ is orthonormal with its range space $\mathrm{range}(\boldsymbol{U})$ containing $\mathcal{S}_t$. Ignoring the role of $\boldsymbol{U}^\top \boldsymbol{Y}_t$ for simplicity, we can write the set $\mathcal{G}_t$ of global minimizers as

$$\mathcal{G}_t := \big\{\boldsymbol{U} : \boldsymbol{U}^\top \boldsymbol{U} = I, \ \mathcal{S}_t \subset \mathrm{range}(\boldsymbol{U}) \subsetneq \mathbb{R}^n\big\};$$

we ruled out the case $\mathrm{range}(\boldsymbol{U}) = \mathbb{R}^n$ as this indicates trivial solutions. Assumption 1 implies the intersection

$$\cap_{i=1}^t \mathcal{G}_i = \{\boldsymbol{U} : \boldsymbol{U}^\top \boldsymbol{U} = I, \sum\nolimits_{i=1}^t \mathcal{S}_i \subset \mathrm{range}(\boldsymbol{U}) \subsetneq \mathbb{R}^n\}$$

is non-empty, which implies $\dim(\sum_{t=1}^T \mathcal{S}_t) < n$ or equivalent that the data matrix $[\boldsymbol{Y}_1 \ \cdots \ \boldsymbol{Y}_T]$ is rank-deficient. Note that this is a weaker assumption than that of Peng & Risteski (2022), who assumed the rank of $[\boldsymbol{Y}_1 \ \cdots \ \boldsymbol{Y}_T]$ is given.

Storing $\mathcal{K}_t$. To implement $\texttt{ICL}^\dagger$ under Assumption 1, we need to store $\mathcal{K}_t = \cap_{i=1}^t \mathcal{G}_i$ in a memory-efficient manner. Since we know that any orthonormal matrix whose range space contains $\sum_{i=1}^t \mathcal{S}_i$ is an element of $\mathcal{K}_t$ (and vice versa), storing $\mathcal{K}_t$ is equivalent to storing the subspace sum $\sum_{i=1}^t \mathcal{S}_i$. Indeed, storing $\sum_i^t \mathcal{S}_i$ gives enough information about $\mathcal{K}_t$, which prevents forgetting (recall the minimality of $\mathcal{K}_t$, §2.3). With this viewpoint, we will maintain an orthonormal basis matrix K_t of $\sum_{i=1}^t \mathcal{S}_i$ to implement $\texttt{ICL}^\dagger$, where each subspace $\mathcal{S}_i$ will be learned from data $\boldsymbol{Y}_i$.

Implementation Details. For $t = 1$, we can compute the orthonormal basis K_1 of $\mathcal{S}_1$ via an SVD[3] on $\boldsymbol{Y}_1$; e.g., set K_1 to be the matrix whose columns are left singular vectors of $\boldsymbol{Y}_1$ corresponding to its non-zero singular values.

For $t > 1$, we need to compute K_t from $\boldsymbol{Y}_t$ and K_{t-1}, under the inductive assumption that K_{t-1} is an orthonormal basis matrix of $\sum_{i=1}^{t-1} \mathcal{S}_i$. Since $\mathcal{K}_{t-1}$ consists of orthonormal matrices $\boldsymbol{U}$ whose range spaces contain $\mathrm{range}(K_{t-1})$, we know that $\boldsymbol{U} \in \mathcal{K}_{t-1}$ if and only if $\boldsymbol{U}^\top \boldsymbol{U} = I$ and $\boldsymbol{U}$ is of the form $[K_{t-1} \ U_t]$ (up to some isometry). As such, the recursion (2) of $\texttt{ICL}^\dagger$ is equivalent to

$$\begin{aligned}
&\operatorname*{argmin}_{\boldsymbol{U} \in \mathcal{K}_{t-1}} \big\|\boldsymbol{U}\boldsymbol{U}^\top \boldsymbol{Y}_t - \boldsymbol{Y}_t\big\|_{\mathrm{F}}^2 \\
\Leftrightarrow\ & \overline{U}_t \in \operatorname*{argmin}_{U_t} \big\|[K_{t-1} \ \ U_t]\,[K_{t-1} \ \ U_t]^\top \boldsymbol{Y}_t - \boldsymbol{Y}_t\big\|_{\mathrm{F}}^2 \\
\Leftrightarrow\ & \overline{U}_t \in \operatorname*{argmin}_{U_t} \big\|U_t U_t^\top Y_t - Y_t\big\|_{\mathrm{F}}
\end{aligned}$$

where we defined $Y_t := (I - K_{t-1}K_{t-1}^\top)\boldsymbol{Y}_t$ and used the fact $K_{t-1}^\top U_t = 0$. Similarly to the case $t = 1$, $\overline{U}_t$ can be computed via an SVD on Y_t, and the rank of Y_t determines the number of its columns. Setting $K_t \leftarrow [K_{t-1} \ \overline{U}_t]$ furnishes a desired orthonormal basis for $\sum_{i=1}^t \mathcal{S}_i$.

Remark 4 (New Knowledge = New Parameters). If $\mathcal{S}_t$ is contained in $\sum_{i=1}^{t-1} \mathcal{S}_i$, then there is no new "knowledge" to learn and $Y_t = 0$; in this case we set $K_t \leftarrow K_{t-1}$. The amount of new knowledge is encoded as the rank of Y_t, which determines the number of new parameters to add.

3.2.3. Connections to Deep Continual Learning

In §3.2.2, $\texttt{ICL}^\dagger$ is shown to be an ***expansion-based method*** that grows the columns of the orthonormal basis K_t adaptively. Recall that the challenges of designing ***expansion-based methods*** include (1) how many parameters to add and (2) where to add them. For the problem of continual matrix factorization, $\texttt{ICL}^\dagger$ addresses these two challenges perfectly by leveraging the eigen structures of features $\boldsymbol{Y}_t$ or projected features Y_t. To our knowledge, no ***expansion-based methods*** in deep continual learning have exploited such structures; this implies an important extension as future work. Also, different from many existing ***expansion-based methods*** (as Yan et al. (2021) reviewed), $\texttt{ICL}^\dagger$ does not require the identity of the task at the test time.

The other important aspect that $\texttt{ICL}^\dagger$ reveals is ***the role of the network width in resisting catastrophic forgetting***. Indeed, the matrix K_t can be viewed as the second layer of a two-layer linear network, and the growth of its columns corresponds to increasing the network width. This principled way of increasing the network width complements the empirical claim of Mirzadeh et al. (2022) and the experimental observation of Rusu et al. (2016); Yoon et al. (2018): *Wide neural networks forget less catastrophically*. Two remarks are in order, to finish the section:

Remark 5 (Grow = Shrink). While $\boldsymbol{K}_t$ *shrinks* in continual linear regression and we have $\mathcal{K}_t \subset \mathcal{K}_{t-1}$ in general, we see that, in continual matrix factorization, K_t *grows* its columns as $\texttt{ICL}^\dagger$ is presented with more subspaces. However, the sum $\sum_{i=1}^t \mathcal{S}_i$ can be uniquely identified with the intersection $\cap_{i=1}^t \mathcal{S}_i^\perp$ of the orthogonal complement $\mathcal{S}_i^\perp$, and therefore if we store a basis matrix for $\cap_{i=1}^t \mathcal{S}_i^\perp$ (instead of for $\sum_{i=1}^t \mathcal{S}_i$), then the memory consumption shrinks over time. We will elaborate on this idea in Appendix G.

Remark 6 (Remember = Forget). While $\mathcal{K}_{t-1}$ *remembers* all common global minimizers, $\texttt{ICL}^\dagger$ actually *forgets* $\mathcal{K}_{t-1}$ as it *never updates* $\mathcal{K}_{t-1}$. Put differently, the ability of a learner to improve the performance on previous tasks with knowledge from new tasks is desired (only) when previous tasks are solved sub-optimally; such ability is typically called *positive backward transfer* (Lin et al., 2022b).

4. Continual Learning Basics: Generalization

In this section we prove that $\texttt{ICL}^\dagger$ is a learner in the *statistical learning* sense (therefore we can remove †); see, e.g., Shalev-Shwartz & Ben-David (2014); Mohri et al. (2018) for some basics of statistical learning. Towards that goal, we now reframe our problem setup in a statistical learning environment. Assume that, for task t, the dataset $D_t = \{\boldsymbol{d}_{ti}\}_{i=1}^{m_t}$ consists of m_t *independent and identically distributed* samples $\boldsymbol{d}_{ti} \overset{\text{i.i.d.}}{\sim} \mathcal{D}_t$, where $\mathcal{D}_t$ is some (unknown) data distribution for task t, and the objective L_t (2) is given as

$$L_t(\boldsymbol{w}; D_t) := \frac{1}{m_t} \sum_{i=1}^{m_t} \ell_t(\boldsymbol{w}; \boldsymbol{d}_{ti}), \tag{11}$$

where $\ell_t(\boldsymbol{w}; \boldsymbol{d}_{ti})$ is the loss on sample $\boldsymbol{d}_{ti}$ evaluated at $\boldsymbol{w}$. Hence, (2) becomes an *empirical risk minimization* problem, and $\texttt{ICL}^\dagger$ becomes a (constrained) empirical risk minimizer.

Corresponding to (11) is the statistical learning task

$$\mathcal{G}_t^* := \operatorname*{argmin}_{\boldsymbol{w} \in \mathcal{W}} \mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\boldsymbol{w}; \boldsymbol{d})]. \tag{12}$$

To proceed, we need the assumption of *shared multitask model* and the Ideal Continual Learner (without †):

Assumption 2 (Shared Multitask Model)**.** $\cap_{t=1}^T \mathcal{G}_t^* \neq \varnothing$.

Definition 2 (The Ideal Continual Learner, $\texttt{ICL}$)**.** With $\mathcal{K}_0^* := \mathcal{W}$, for $t = 1, 2, \ldots, T$, the algorithm $\texttt{ICL}$ solves

$$\mathcal{K}_t^* \leftarrow \operatorname*{argmin}_{\boldsymbol{w} \in \mathcal{K}_{t-1}^*} \mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\boldsymbol{w}; \boldsymbol{d})]. \tag{13}$$

Much of what we said to $\texttt{ICL}^\dagger$ and Assumption 1 applies to the Ideal Continual Learner ($\texttt{ICL}$) and Assumption 2; we shall not repeat here. Instead, we will investigate the possibility of minimizing (12) or (13) via $\texttt{ICL}^\dagger$.

One difficulty of such investigation is as follows. While $\mathcal{G}_t$ of (1) approaches $\mathcal{G}_t^*$ of (12) as sample size m_t tends to infinity, in general we have $\mathcal{G}_t \neq \mathcal{G}_t^*$ when m_t is finite. As such, Assumptions 1 and 2 are different (though related), and, in general, one of them does not imply the other. As a consequence, the sufficiency of $\texttt{ICL}$ under Assumption 2 does not imply the sufficiency of $\texttt{ICL}^\dagger$, which requires Assumption 1 (cf. Proposition 1); and vice versa. This difficulty disappears if we make both Assumptions 1 and 2, which, however, would ask for too much.

We will proceed with either Assumption 1 (§4.1) or Assumption 2 (§4.2), but not both. To do so, we need some assumptions on the objective functions L_t and hypothesis space $\mathcal{W}$, which are standard in statistical learning:

Assumption 3 (Uniform Convergence)**.** Let B and M be two positive constants. Assume $\|\boldsymbol{w}\|_2 \leq B$ for every $\boldsymbol{w} \in \mathcal{W}$. Assume that $\ell_t(\boldsymbol{w}; \boldsymbol{d})$ is M-Lipschitz in $\boldsymbol{w}$ for every sample $\boldsymbol{d} \sim \mathcal{D}_t$ and every $t = 1, \ldots, T$, i.e.,

$$|\ell_t(\boldsymbol{w}; \boldsymbol{d}) - \ell_t(\boldsymbol{w}'; \boldsymbol{d})| \leq M \cdot \|\boldsymbol{w} - \boldsymbol{w}'\|_2, \ \forall \boldsymbol{w}, \boldsymbol{w}' \in \mathcal{W}.$$

These assumptions suffice to ensure *uniform convergence* (Shalev-Shwartz et al., 2009, Thm 5): For fixed t and $\delta \in (0, 1)$, with probability at least $1 - \delta$ we have ($\forall \boldsymbol{w} \in \mathcal{W}$)

$$|L_t(\boldsymbol{w}; D_t) - \mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\boldsymbol{w}; \boldsymbol{d})]| \leq \zeta(m_t, \delta), \tag{14}$$

where $\zeta(m_t, \delta) := O\left(\frac{MB\sqrt{n \log(m_t) \log(n/\delta)}}{\sqrt{m_t}}\right)$

approaches zero as the sample size m_t tends to infinity.

The term $\zeta(m_t, \delta)$ defined in (14) serves as a uniform convergence bound and will appear frequently in our generalization bounds. While $\zeta(m_t, \delta)$ approaches zero as the sample size m_t tends to infinity (assuming other terms are fixed), it is also controlled by other factors as (14) suggests. First of all, it depends on the failure probability δ, and we would like to set δ small. Second, it depends on the Lipschitz continuity constant M of the loss function; in the context of deep learning, M is controlled by network architectures and its value is typically unknown, although it might be estimated in certain cases via computationally intensive algorithms (Fazlyab et al., 2019). Third, $\zeta(m_t, \delta)$ also depends on the dimension n of the variable $\boldsymbol{w}$ (e.g., the number of parameters in a deep network). In particular, we have $\zeta(m_t, \delta) = O(\sqrt{n \log n / m_t})$ (ignoring other parameters), and so, for $\zeta(m_t, \delta)$ to be small, we need $m_t \gg n \log n$.

4.1. ICL† Learns All Tasks

In this section, we proceed with Assumption 1 and without Assumption 2. Our main result is this (see also Figure 3):

Theorem 1 (ICL† $\Rightarrow$ All-Task Learner)**.** *Let c_t^* be the minimum value of (12). Let $\hat{\boldsymbol{w}} \in \mathcal{K}_T$. Let $\delta \in (0,1)$ and $\zeta(m_t, \delta)$ be as in Assumption 3. Under Assumptions 1 and 3, with probability at least $1-\delta$, we have ($\forall t = 1, \dots, T$)*

$$c_t^* \leq \mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\hat{\boldsymbol{w}}; \boldsymbol{d})] \leq c_t^* + \zeta(m_t, \delta/T). \tag{15}$$

As (14) implies, an exponentially large T would make the numerator of the error term $\zeta(m_t, \delta/T)$ in (15) large, while a large number of samples m_t would make it small. In the extreme case $T \to \infty$, it is necessary to have $m_t \to \infty$, otherwise $\zeta(m_t, \delta/T)$ would be infinity and the upper bound of (15) would be invalid. In the current deep continual learning practice, though, T is of the constant order, e.g., $T \leq 1000$ (Lesort et al., 2022). Note then that, for fixed T and δ, as the sample size m_t tends to ∞ for every $t = 1, \dots, T$, we have $\zeta(m_t, \delta/T) \to 0$, in which case ICL† becomes an all-task learner: Every point $\hat{\boldsymbol{w}}_T$ of $\mathcal{K}_T$ that it finds reaches the minimum values of all learning tasks (12).

Remark 7 (Approximate Sufficiency). Assumptions 1 and 3 promise $\hat{\boldsymbol{w}}$ (15) to be an *approximate* global minimizer to all learning tasks (12); or we might say that they make ICL *approximately sufficient* even without Assumption 2.

4.2. Relaxed Continual Learner† = ICL

Here, we consider Assumption 2, at the risk that ICL† might not be sufficient. However, by deriving generalization bounds, we will prove that a *relaxation* of ICL† is ICL.

While ICL† can perform arbitrarily worse for certain pathological cases in the absence of Assumption 1 (as Example 1 showed), Assumptions 2 and 3 come into play, making ICL† *approximately sufficient* (similarly to Remark 7):

Proposition 7 (Approximate Sufficiency)**.** *Let $\delta \in (0,1)$. Assumptions 2, 3 imply there is a point $\boldsymbol{w} \in \mathcal{W}$ satisfying*

$$L_t(\boldsymbol{w}; D_t) \leq c_t + \zeta(m_t, \delta/T), \quad \forall t = 1, \dots, T$$

with probability at least $1-\delta$. Here c_t and $\zeta(m_t, \delta)$ are respectively defined in Proposition 2 and Assumption 3.

In the absence of Assumption 1, the constraint of ICL† in Proposition 2 might be prohibitive and lead to sub-optimal solutions (cf. Example 1). This is why we relax it as per Proposition 7 by some additive factor:

Definition 3 (Relaxed Continual Learner†)**.** With $\mathcal{K}_0 := \mathcal{W}$, the *relaxed* Continual Learner† is an algorithm that solves the following program sequentially for $t = 1, 2, \dots, T$:

$$\min_{\boldsymbol{w} \in \mathcal{W}} L_t(\boldsymbol{w}; D_t) \tag{16}$$
$$\text{s.t. } L_i(\boldsymbol{w}; D_i) \leq c_i + \zeta(m_i, \delta/t), \ \forall i = 1, \dots, t-1$$

We can now state the following (see also Figure 3):

Theorem 2 (Relaxed Continual Learner† $\approx$ ICL)**.** *Let $\delta \in (0,1)$. Let c_t^* be the minimum of (12) and $\zeta(m_t, \delta)$ defined in (14). Suppose Assumptions 2 and 3 hold. With probability at least $1-\delta$, the relaxed Continual Learner† is feasible, and its global minimizer $\overline{\boldsymbol{w}}$ satisfies ($\forall t = 1, \dots, T$)*

$$c_t^* \leq \mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\overline{\boldsymbol{w}}; \boldsymbol{d})] \leq c_t^* + \zeta(m_t, \delta/T). \tag{17}$$

4.3. ICL and Constrained Learning

Similarly to Proposition 2, the constraint $\boldsymbol{w} \in \mathcal{K}_{t-1}^*$ of ICL can be written as inequality constraints $\mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_i}[\ell_i(\boldsymbol{w}; \boldsymbol{d})] \leq c_i^*$ ($\forall i = 1, \dots, t-1$), where c_i^* is the minimum for task i (12). If the values of c_i^* are instead determined by applications (not as minima of prior tasks), then the resulting learning problem coincides with *constrained learning* (Chamon et al., 2022). Constrained learning naturally arises when learning under certain requirements, e.g., robustness (Zhang et al., 2019; Robey et al., 2021), safety (Paternain et al., 2019), or fairness (Cotter et al., 2019). In a different line of research, constrained learning is also called *stochastic optimization with expectation constraints* (Yu et al., 2017; Bedi et al., 2019; Madavan & Bose, 2021; Akhtar et al., 2021). For both problems, several algorithms have recently emerged with theoretical guarantees. For example, Chamon et al. (2022) solve the constrained learning problem using Lagrangian multipliers, which can be viewed as a ***regularization-based method*** as per a common deep continual learning taxonomy. Left though to future work, leveraging this line of research might be key to improving the current continual learning systems. Finally, note that Chamon et al. (2022) developed generalization bounds for constrained learning via different mathematical mechanisms (e.g., primal-dual analysis), and we refer the reader to their works for a detailed exposition of these ideas.

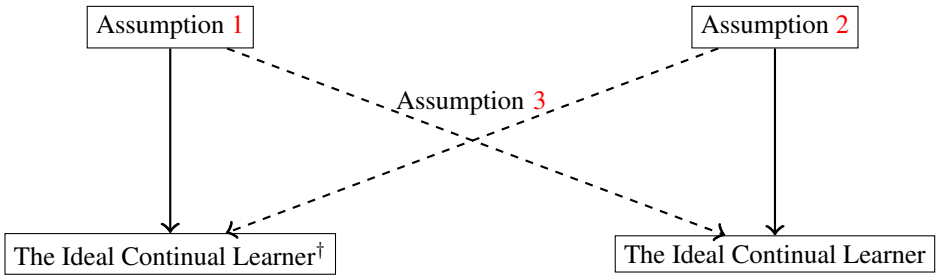


Figure 3: Figure 3 is designed to help the reader understand our results of §4.1 and §4.2 at a high level, and it can be read as follows. Assumptions 1 and 2 guarantee the sufficiency of the Ideal Continual Learner† and the Ideal Continual Learner, respectively, and this is denoted by solid arrows in the figure. Assumptions 1 and 3 indicate an approximate sufficiency of the Ideal Continual Learner† (cf. Theorem 1 and Remark 7), and this is denoted by a dashed arrow in the figure.

4.4. ICL and Rehearsal

Using the ICL framework, we can now acquire a theoretical understanding of research and debate (Q2). For each task $t = 1, \ldots, T-1$, assume we stored s_t samples (out of m_t), and we now face task T. We *rehearse*, i.e., retrain with all stored samples and data of task T, to minimize

$$\min_{\boldsymbol{w} \in \mathcal{W}} \frac{1}{m_T} \sum_{i=1}^{m_T} \ell_T(\boldsymbol{w}; \boldsymbol{d}_{Ti}) + \sum_{t=1}^{T-1} \frac{1}{s_t} \sum_{i=1}^{s_t} \ell_t(\boldsymbol{w}; \boldsymbol{d}_{ti}). \quad (18)$$

Rehearsal has shown good empirical performance in deep continual learning; see, e.g., Chaudhry et al. (2019); Prabhu et al. (2020); Zhang et al. (2022); Bonicelli et al. (2022). For the first time to our knowledge, a theoretical justification that accounts for its effectiveness is provided here:

Theorem 3 (Rehearsal $\approx$ ICL)**.** *Under Assumptions 2 and 3, with probability at least $1-\delta$, every global minimizer $\overline{\boldsymbol{w}}$ of the rehearsal objective (18) satisfies*

$$\sum_{t=1}^{T} c_t^* \leq \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\overline{\boldsymbol{w}}; \boldsymbol{d})] \leq \sum_{t=1}^{T} c_t^* + \text{ERROR}$$
$$\text{where ERROR} := \zeta(m_T, \delta/T) + \sum_{t=1}^{T-1} \zeta(s_t, \delta/T). \quad (19)$$

Note how (19) reveals the way rehearsal affects generalization: A larger memory buffer (s_t) means a smaller $\zeta(s_t, \delta/T)$, implying better generalization. Crucially, this bound remains the same as long as we store the same number of samples for each task (say s_t) ***regardless of which samples we store*** (a natural consequence of the i.i.d. assumption). This deviates from the trendy idea of selecting *representative* samples in recent methods—reportedly, these methods do not necessarily outperform a simple random selection mechanism (Araujo et al., 2022). Due to space, more elaborations are put in Appendix B.2.

We can then compare the relaxed Continual Learner† (§4.2) and rehearsal (§4.4). Note first that one could also implement the relaxed Continual Learner† using a subset of samples similar to rehearsal. Theoretically, the relaxed Continual Learner† enjoys stronger generalization guarantees than rehearsal: Theorem 2 bounds generalization errors for individual tasks, while Theorem 3 only gives a bound on the multitask error. Computationally, rehearsal is simpler to implement than the relaxed Continual Learner† and can maintain good performance (Chaudhry et al., 2019; Prabhu et al., 2020; Zhang et al., 2022; Bonicelli et al., 2022).

5. Conclusion

Under the ICL framework, we derived and justified many existing methods in deep continual learning. Put conversely, *all roads lead to Rome*: Many prior continual learning methods developed with different insights and different motivations turn out to be (approximately) ICL. Importantly, we made several connections to other research fields. This activates opportunities for improving the existing (deep) continual learning systems, for which we venture to hope that ICL serves as a primary design principle.

The ICL framework in its present form comes with limitations. On the theoretical side, we tacitly assumed that the samples of each task t are drawn i.i.d. from distribution $\mathcal{D}_t$ in our generalization theory. While this i.i.d. assumption is standard in classic statistical learning, dispensing with it and accounting for out-of-distribution data within tasks is left as future work. On the algorithmic front, we emphasize that, in general, and particularly for deep continual learning, it is by no means easy to develop an *exact* implementation of ICL, which constitutes a major limitation of the proposed framework. Nevertheless, our paper has suggested many possibilities for approximating the Ideal Continual Learner. We believe devising such approximations with problem-specific insights will be a promising research direction.

Acknowledgements. This work is supported by the project ULEARN "Unsupervised Lifelong Learning" and co-funded under the grant number 316080 of the Research Council of Norway.

References

Akhtar, Z., Bedi, A. S., and Rajawat, K. Conservative stochastic optimization with expectation constraints. *IEEE Transactions on Signal Processing*, 69:3190–3205, 2021.

Aljundi, R., Babiloni, F., Elhoseiny, M., Rohrbach, M., and Tuytelaars, T. Memory aware synapses: Learning what (not) to forget. In *European Conference on Computer Vision*, 2018.

Aljundi, R., Kelchtermans, K., and Tuytelaars, T. Task-free continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019a.

Aljundi, R., Lin, M., Goujaud, B., and Bengio, Y. Gradient based sample selection for online continual learning. In *Advances in Neural Information Processing Systems*, 2019b.

Araujo, V., Balabin, H., Hurtado, J., Soto, A., and Moens, M.-F. How relevant is selective memory population in lifelong language learning? In *Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 2022.

Asanuma, H., Takagi, S., Nagano, Y., Yoshida, Y., Igarashi, Y., and Okada, M. Statistical mechanical analysis of catastrophic forgetting in continual learning with teacher and student networks. *Journal of the Physical Society of Japan*, 90(10):104001, 2021.

Balcan, M.-F., Blum, A., and Vempala, S. Efficient representations for lifelong learning and autoencoding. In *Conference on Learning Theory*, pp. 191–210, 2015.

Baldi, P. and Hornik, K. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2(1):53–58, 1989.

Balzano, L., Chi, Y., and Lu, Y. M. Streaming PCA and subspace tracking: The missing data case. *Proceedings of the IEEE*, 106(8):1293–1310, 2018.

Bang, J., Kim, H., Yoo, Y., Ha, J.-W., and Choi, J. Rainbow memory: Continual learning with a memory of diverse samples. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Baxter, J. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000.

Bedi, A. S., Koppel, A., and Rajawat, K. Asynchronous saddle point algorithm for stochastic optimization in heterogeneous networks. *IEEE Transactions on Signal Processing*, 67(7):1742–1757, 2019.

Ben-David, S. and Borbely, R. S. A notion of task relatedness yielding provable multiple-task learning guarantees. *Machine Learning*, 73(3):273–287, 2008.

Bennani, M. A., Doan, T., and Sugiyama, M. Generalisation guarantees for continual learning with orthogonal gradient descent. Technical report, arXiv:2006.11942v4 [stat.ML], 2020.

Bonicelli, L., Boschini, M., Porrello, A., Spampinato, C., and Calderara, S. On the effectiveness of Lipschitz-driven rehearsal in continual learning. In *Advances in Neural Information Processing Systems*, 2022.

Borsos, Z., Mutny, M., and Krause, A. Coresets via bilevel optimization for continual learning and streaming. *Advances in Neural Information Processing Systems*, 2020.

Borwein, J. M. and Tam, M. K. A cyclic Douglas–Rachford iteration scheme. *Journal of Optimization Theory and Applications*, 160(1):1–29, 2014.

Boyle, J. P. and Dykstra, R. L. A method for finding projections onto the intersection of convex sets in Hilbert spaces. In *Advances in Order Restricted Statistical Inference*, 1986.

Brand, M. Incremental singular value decomposition of uncertain data with missing values. In *European Conference on Computer Vision*, 2002.

Brand, M. Fast low-rank modifications of the thin singular value decomposition. *Linear Algebra and its Applications*, 415(1):20–30, 2006.

Bunch, J. R. and Nielsen, C. P. Updating the singular value decomposition. *Numerische Mathematik*, 31(2):111–129, 1978.

Buzzega, P., Boschini, M., Porrello, A., Abati, D., and Calderara, S. Dark experience for general continual learning: a strong, simple baseline. *Advances in Neural Information Processing Systems*, 2020.

Cao, X., Liu, W., and Vempala, S. Provable lifelong learning of representations. In *International Conference on Artificial Intelligence and Statistics*, 2022.

Chamon, L. F., Paternain, S., Calvo-Fullana, M., and Ribeiro, A. Constrained learning with non-convex losses. *IEEE Transactions on Information Theory*, 2022.

Chaudhry, A., Rohrbach, M., Elhoseiny, M., Ajanthan, T., Dokania, P. K., Torr, P. H., and Ranzato, M. On tiny episodic memories in continual learning. Technical report, arXiv:1902.10486v4 [cs.LG], 2019.

Chaudhry, A., Khan, N., Dokania, P., and Torr, P. Continual learning in low-rank orthogonal subspaces. *Advances in Neural Information Processing Systems*, 2020.

Chen, X., Papadimitriou, C., and Peng, B. Memory bounds for continual learning. In *IEEE Symposium on Foundations of Computer Science*, 2022.

Cimmino, G. Calcolo approssimato per le soluzioni dei sistemi di equazioni lineari. *La Ricerca Scientifica (Roma)*, pp. 236–333, 1938.

Combettes, P. L. The foundations of set theoretic estimation. *Proceedings of the IEEE*, 81(2):182–208, 1993.

Cotter, A., Jiang, H., Gupta, M., Wang, S., Narayan, T., You, S., and Sridharan, K. Optimization with non-differentiable constraints with applications to fairness, recall, churn, and other goals. *Journal of Machine Learning Research*, 20(172):1–59, 2019.

De Lange, M., Aljundi, R., Masana, M., Parisot, S., Jia, X., Leonardis, A., Slabaugh, G., and Tuytelaars, T. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3366–3385, 2022.

Dempe, S. and Dutta, J. Is bilevel programming a special case of a mathematical program with complementarity constraints? *Mathematical Programming*, 131(1):37–48, 2012.

Doan, T., Bennani, M. A., Mazoure, B., Rabusseau, G., and Alquier, P. A theoretical analysis of catastrophic forgetting through the NTK overlap matrix. In *International Conference on Artificial Intelligence and Statistics*, 2021.

Douillard, A., Ramé, A., Couairon, G., and Cord, M. DyTox: Transformers for continual learning with dynamic token expansion. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Du, S. S., Hu, W., Kakade, S. M., Lee, J. D., and Lei, Q. Few-shot learning via learning the representation, provably. In *International Conference on Learning Representations*, 2021.

Evron, I., Moroshko, E., Ward, R., Srebro, N., and Soudry, D. How catastrophic can catastrophic forgetting be in linear regression? In *Conference on Learning Theory*, 2022.

Evron, I., Moroshko, E., Buzaglo, G., Khriesh, M., Marjieh, B., Srebro, N., and Soudry, D. Continual learning in linear classification on separable data. In *International Conference on Machine Learning*, 2023.

Farajtabar, M., Azizan, N., Mott, A., and Li, A. Orthogonal gradient descent for continual learning. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Fazlyab, M., Robey, A., Hassani, H., Morari, M., and Pappas, G. Efficient and accurate estimation of Lipschitz constants for deep neural networks. *Advances in Neural Information Processing Systems*, 2019.

Fisher, R. A. On the mathematical foundations of theoretical statistics. *Philosophical Transactions of the Royal Society of London*, 222(594-604):309–368, 1922.

Frascaroli, E., Benaglia, R., Boschini, M., Moschella, L., Fiorini, C., Rodolà, E., and Calderara, S. CaSpeR: Latent spectral regularization for continual learning. Technical report, arXiv:2301.03345 [cs.LG], 2023.

Guo, Y., Hu, W., Zhao, D., and Liu, B. Adaptive orthogonal projection for batch and online continual learning. *AAAI Conference on Artificial Intelligence*, 2022.

Hayes, T. L. and Kanan, C. Selective replay enhances learning in online continual analogical reasoning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Haykin, S. S. *Adaptive Filter Theory*. Pearson Education India, 2002.

Heckel, R. Provable continual learning via sketched Jacobian approximations. In *International Conference on Artificial Intelligence and Statistics*, 2022.

Hung, C.-Y., Tu, C.-H., Wu, C.-E., Chen, C.-H., Chan, Y.-M., and Chen, C.-S. Compacting, picking and growing for unforgetting continual learning. *Advances in Neural Information Processing Systems*, 2019.

Isele, D. and Cosgun, A. Selective experience replay for lifelong learning. In *AAAI Conference on Artificial Intelligence*, 2018.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in Neural Information Processing Systems*, 2018.

Jiang, R., Abolfazli, N., Mokhtari, A., and Yazdandoost Hamedani, E. A conditional gradient-based method for simple bilevel optimization with convex lower-level problem. Technical report, arXiv:2206.08868v2 [math.OC], 2022.

Jin, X., Sadhu, A., Du, J., and Ren, X. Gradient-based editing of memory examples for online task-free continual learning. *Advances in Neural Information Processing Systems*, 2021.

Jolliffe, I. *Principal Component Analysis (2nd ed.)*. Springer, 2002.

Karczmarz, S. Angenäherte auflösung von systemen linearer gleichungen. *Bullet de l'Académie des Sciences de Pologne*, pp. 355–357, 1937.

Kieffer, M., Jaulin, L., and Walter, E. Guaranteed recursive nonlinear state estimation using interval analysis. In *IEEE Conference on Decision and Control*, 1998.

Kim, G., Liu, B., and Ke, Z. A multi-head model for continual learning via out-of-distribution replay. In *Conference on Lifelong Learning Agents*, 2022.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of The National Academy of Sciences*, 114(13):3521–3526, 2017.

Knoblauch, J., Husain, H., and Diethe, T. Optimal continual learning has perfect memory and is NP-hard. In *International Conference on Machine Learning*, 2020.

Kong, Y., Liu, L., Wang, Z., and Tao, D. Balancing stability and plasticity through advanced null space in continual learning. In *European Conference on Computer Vision*, 2022.

Lee, S., Ha, J., Zhang, D., and Kim, G. A neural Dirichlet process mixture model for task-free continual learning. In *International Conference on Learning Representations*, 2020.

Lee, S., Goldt, S., and Saxe, A. Continual learning in the teacher-student setup: Impact of task similarity. In *International Conference on Machine Learning*, 2021.

Lehmann, E. L. and Scheffé, H. Completeness, similar regions, and unbiased estimation: Part i. *Sankhyā: The Indian Journal of Statistics*, 10(4):305–340, 1950.

Lerman, G., McCoy, M. B., Tropp, J. A., and Zhang, T. Robust computation of linear models by convex relaxation. *Foundations of Computational Mathematics*, 15 (2):363–410, 2015.

Lesort, T., Stoian, A., and Filliat, D. Regularization shortcomings for continual learning. Technical report, arXiv:1912.03049v4 [cs.LG], 2019.

Lesort, T., Ostapenko, O., Misra, D., Arefin, M. R., Rodríguez, P., Charlin, L., and Rish, I. Scaling the number of tasks in continual learning. Technical report, arXiv:2207.04543 [cs.LG], 2022.

Li, X., Zhou, Y., Wu, T., Socher, R., and Xiong, C. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International Conference on Machine Learning*, 2019.

Li, Y., Li, M., Asif, M. S., and Oymak, S. Provable and efficient continual representation learning. Technical report, arXiv:2203.02026v2 [cs.LG], 2022.

Lin, S., Yang, L., Fan, D., and Zhang, J. TRGP: Trust region gradient projection for continual learning. In *International Conference on Learning Representations*, 2022a.

Lin, S., Yang, L., Fan, D., and Zhang, J. Beyond not-forgetting: Continual learning with backward knowledge transfer. In *Advances in Neural Information Processing Systems*, 2022b.

Lindstrom, S. B. and Sims, B. Survey: Sixty years of Douglas–Rachford. *Journal of the Australian Mathematical Society*, 110(3):333–370, 2021.

Liu, B., Liu, Q., and Stone, P. Continual learning and private unlearning. In *Conference on Lifelong Learning Agents*, 2022.

Liu, H. and Liu, H. Continual learning with recursive gradient optimization. In *International Conference on Learning Representations*, 2022.

Liu, X. Awesome incremental learning / lifelong learning. http://https://github.com/xialeiliu/Awesome-Incremental-Learning, 2022. Accessed: August 2022.

Liu, X., Masana, M., Herranz, L., Van de Weijer, J., Lopez, A. M., and Bagdanov, A. D. Rotate your networks: Better weight consolidation and less catastrophic forgetting. In *International Conference on Pattern Recognition*, 2018.

Lopez-Paz, D. and Ranzato, M. Gradient episodic memory for continual learning. *Advances in Neural Information Processing Systems*, 2017.

Madavan, A. N. and Bose, S. A stochastic primal-dual method for optimization with conditional value at risk constraints. *Journal of Optimization Theory and Applications*, 190(2):428–460, 2021.

Mallya, A. and Lazebnik, S. PackNet: Adding multiple tasks to a single network by iterative pruning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

Maurer, A., Pontil, M., and Romera-Paredes, B. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32, 2016.

McCloskey, M. and Cohen, N. J. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of Learning and Motivation*, volume 24, pp. 109–165. 1989.

Mirzadeh, S. I., Farajtabar, M., Gorur, D., Pascanu, R., and Ghasemzadeh, H. Linear mode connectivity in multitask and continual learning. In *International Conference on Learning Representations*, 2021.

Mirzadeh, S. I., Chaudhry, A., Yin, D., Hu, H., Pascanu, R., Gorur, D., and Farajtabar, M. Wide neural networks forget less catastrophically. In *International Conference on Machine Learning*, 2022.

Mitliagkas, I., Caramanis, C., and Jain, P. Memory limited, streaming PCA. *Advances in Neural Information Processing Systems*, 2013.

Mohri, M., Rostamizadeh, A., and Talwalkar, A. *Foundations of Machine Learning*. MIT Press, 2018.

Narayanamurthy, P. and Vaswani, N. Provable dynamic robust PCA or robust subspace tracking. *IEEE Transactions on Information Theory*, 65(3):1547–1577, 2018.

Nesterov, Y. *Lectures on Convex Optimization*. Springer, 2018.

Oja, E. Simplified neuron model as a principal component analyzer. *Journal of Mathematical Biology*, 15(3):267–273, 1982.

Parisi, G. I., Kemker, R., Part, J. L., Kanan, C., and Wermter, S. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Park, D., Hong, S., Han, B., and Lee, K. M. Continual learning by asymmetric loss approximation with single-side overestimation. In *IEEE/CVF International Conference on Computer Vision*, 2019.

Paternain, S., Chamon, L., Calvo-Fullana, M., and Ribeiro, A. Constrained reinforcement learning has zero duality gap. *Advances in Neural Information Processing Systems*, 2019.

Pearson, K. On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11): 559–572, 1901.

Peng, B. and Risteski, A. Continual learning: A feature extraction formalization, an efficient algorithm, and fundamental obstructions. In *Advances in Neural Information Processing Systems*, 2022.

Pourcel, J., Vu, N.-S., and French, R. M. Online task-free continual learning with dynamic sparse distributed memory. In *European Conference on Computer Vision*, 2022.

Prabhu, A., Torr, P. H., and Dokania, P. K. GDumb: A simple approach that questions our progress in continual learning. In *European Conference on Computer Vision*, 2020.

Prado, D. B. and Riddle, P. A theory for knowledge transfer in continual learning. In *Conference on Lifelong Learning Agents*, 2022.

Qu, H., Rahmani, H., Xu, L., Williams, B., and Liu, J. Recent advances of continual learning in computer vision: An overview. *arXiv:2109.11369 [cs.CV]*, 2021.

Ramesh, R. and Chaudhari, P. Model zoo: A growing brain that learns continually. In *International Conference on Learning Representations*, 2022.

Rebuffi, S.-A., Kolesnikov, A., Sperl, G., and Lampert, C. H. iCaRL: Incremental classifier and representation learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., , and Tesauro, G. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *International Conference on Learning Representations*, 2019.

Ritter, H., Botev, A., and Barber, D. Online structured laplace approximations for overcoming catastrophic forgetting. *Advances in Neural Information Processing Systems*, 31, 2018.

Robey, A., Chamon, L., Pappas, G. J., Hassani, H., and Ribeiro, A. Adversarial robustness with semi-infinite constrained learning. *Advances in Neural Information Processing Systems*, 2021.

Robins, A. Catastrophic forgetting in neural networks: The role of rehearsal mechanisms. In *The First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, 1993.

Rusu, A. A., Rabinowitz, N. C., Desjardins, G., Soyer, H., Kirkpatrick, J., Kavukcuoglu, K., Pascanu, R., and Hadsell, R. Progressive neural networks. Technical report, arXiv:1606.04671v3 [cs.LG], 2016.

Saha, G. and Roy, K. Saliency guided experience packing for replay in continual learning. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023.

Saha, G., Garg, I., and Roy, K. Gradient projection memory for continual learning. In *International Conference on Learning Representations*, 2021.

Shaheen, K., Hanif, M. A., Hasan, O., and Shafique, M. Continual learning for real-world autonomous systems: Algorithms, challenges and frameworks. *Journal of Intelligent & Robotic Systems*, 105(1):1–32, 2022.

Shalev-Shwartz, S. and Ben-David, S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

Shalev-Shwartz, S., Shamir, O., Srebro, N., and Sridharan, K. Stochastic convex optimization. In *Conference on Learning Theory*, 2009.

Shibata, T., Irie, G., Ikami, D., and Mitsuzumi, Y. Learning with selective forgetting. In *International Joint Conference on Artificial Intelligence*, 2021.

Shin, H., Lee, J. K., Kim, J., and Kim, J. Continual learning with deep generative replay. *Advances in Neural Information Processing Systems*, 2017.

Singhal, S. and Wu, L. Training feed-forward networks with the extended kalman algorithm. In *International Conference on Acoustics, Speech, and Signal Processing*, 1989.

Sun, S., Calandriello, D., Hu, H., Li, A., and Titsias, M. Information-theoretic online memory selection for continual learning. In *International Conference on Learning Representations*, 2022.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. MIT Press, 2018.

Theodoridis, S., Slavakis, K., and Yamada, I. Adaptive learning in a world of projections. *IEEE Signal Processing Magazine*, 28(1):97–123, 2010.

Thrun, S. and Mitchell, T. M. Lifelong robot learning. *Robotics and Autonomous Systems*, 15(1-2):25–46, 1995.

Tiwary, H. R. On the hardness of computing intersection, union and minkowski sum of polytopes. *Discrete & Computational Geometry*, 40(3):469–479, 2008.

Trefethen, L. N. and Bau, D. *Numerical Linear Algebra*. SIAM, 1997.

Tripuraneni, N., Jin, C., and Jordan, M. Provable meta-learning of linear representations. In *International Conference on Machine Learning*, 2021.

Tsakiris, M. C. and Vidal, R. Dual principal component pursuit. *Journal of Machine Learning Research*, 19(18): 1–50, 2018.

van de Ven, G. M., Tuytelaars, T., and Tolias, A. S. Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197, 2022.

Vaswani, N., Bouwmans, T., Javed, S., and Narayanamurthy, P. Robust subspace learning: Robust PCA, robust subspace tracking, and robust subspace recovery. *IEEE Signal Processing Magazine*, 35(4):32–55, 2018.

Verwimp, E., De Lange, M., and Tuytelaars, T. Rehearsal revealed: The limits and merits of revisiting samples in continual learning. In *IEEE/CVF International Conference on Computer Vision*, 2021.

Vicente, L. N. and Calamai, P. H. Bilevel and multilevel programming: A bibliography review. *Journal of Global Optimization*, 5(3):291–306, 1994.

Vidal, R. Mathematics of deep learning. https://www.cis.jhu.edu/~rvidal/talks/learning/DeepMath2020.pdf, 2020. Accessed: January 20, 2023.

Vidal, R., Ma, Y., and Sastry, S. *Generalized Principal Component Analysis*. Springer, 2016.

Vitter, J. S. Random sampling with a reservoir. *ACM Transactions on Mathematical Software*, 11(1):37–57, 1985.

Wang, L., Zhang, M., Jia, Z., Li, Q., Bao, C., Ma, K., Zhu, J., and Zhong, Y. AFEC: Active forgetting of negative transfer in continual learning. *Advances in Neural Information Processing Systems*, 2021a.

Wang, L., Zhang, X., Yang, K., Yu, L., Li, C., Hong, L., Zhang, S., Li, Z., Zhong, Y., and Zhu, J. Memory replay with data compression for continual learning. In *International Conference on Learning Representations*, 2022a.

Wang, S., Li, X., Sun, J., and Xu, Z. Training networks in null space of feature covariance for continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021b.

Wang, Z., Liu, L., Duan, Y., Kong, Y., and Tao, D. Continual learning with lifelong vision transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022b.

Wang, Z., Shen, L., Fang, L., Suo, Q., Duan, T., and Gao, M. Improving task-free continual learning by distributionally robust memory evolution. In *International Conference on Machine Learning*, 2022c.

Witsenhausen, H. Sets of possible states of linear systems given perturbed observations. *IEEE Transactions on Automatic Control*, 13(5):556–558, 1968.

Wu, Z., Tran, H., Pirsiavash, H., and Kolouri, S. Is multi-task learning an upper bound for continual learning? Technical report, arXiv:2210.14797 [cs.LG], 2022.

Xie, J., Yan, S., and He, X. General incremental learning with domain-aware categorical representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Yan, S., Xie, J., and He, X. DER: Dynamically expandable representation for class incremental learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

Yang, B. Projection approximation subspace tracking. *IEEE Transactions on Signal Processing*, 43(1):95–107, 1995.

Yang, B. Asymptotic convergence analysis of the projection approximation subspace tracking algorithms. *Signal Processing*, 50(1-2):123–136, 1996.

Ye, F. and Bors, A. G. Task-free continual learning via online discrepancy distance learning. In *Advances in Neural Information Processing Systems*, 2022.

Ye, T. and Du, S. S. Global convergence of gradient descent for asymmetric low-rank matrix factorization. *Advances in Neural Information Processing Systems*, 2021.

Yin, D., Farajtabar, M., Li, A., Levine, N., and Mott, A. Optimization and generalization of regularization-based continual learning: a loss approximation viewpoint. Technical report, arXiv:2006.10974v3 [cs.LG], 2020.

Yoon, J., Yang, E., Lee, J., and Hwang, S. J. Lifelong learning with dynamically expandable networks. In *International Conference on Learning Representations*, 2018.

Yu, H., Neely, M., and Wei, X. Online convex optimization with stochastic constraints. *Advances in Neural Information Processing Systems*, 2017.

Zeng, G., Chen, Y., Cui, B., and Yu, S. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1(8):364–372, 2019.

Zenke, F., Poole, B., and Ganguli, S. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, 2017.

Zeno, C., Golan, I., Hoffer, E., and Soudry, D. Task agnostic continual learning using online variational bayes. Technical report, arXiv:1803.10123v3 [stat.ML], 2018.

Zeno, C., Golan, I., Hoffer, E., and Soudry, D. Task-agnostic continual learning using online variational bayes with fixed-point updates. *Neural Computation*, 33(11):3139–3177, 2021.

Zhang, H., Yu, Y., Jiao, J., Xing, E., El Ghaoui, L., and Jordan, M. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, 2019.

Zhang, Y., Pfahringer, B., Frank, E., Bifet, A., Lim, N. J. S., and Jia, A. A simple but strong baseline for online continual learning: Repeated augmented rehearsal. In *Advances in Neural Information Processing Systems*, 2022.

Zhou, M., Xiao, J., Chang, Y., Fu, X., Liu, A., Pan, J., and Zha, Z.-J. Image de-raining via continual learning. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021.

A. Structure of The Appendix

We structure the appendix as follows:

- In Appendix B, we elaborate on several points for the reader to better understand and appreciate the main paper.
- In Appendix C, we prove Proposition 6, showing that OGD is approximately the Ideal Continual Learner† in the case of continual linear regression.
- In Appendix D, we review related works on continual learning, emphasizing deep continual learning methods that we mentioned in Remark 3, existing assumptions on task relationships, and existing theoretical papers.
- We review related works in other related fields that are visualized in Figure 1. The emphasis is put on their connections to the Ideal Continual Learner and on detailed comparisons of our approach to related works.
  - In Appendix E, we review related works in set-theoretical estimation. This allows us to understand the Ideal Continual Learner from different perspectives.
  - In Appendix F, we review streaming PCA, incremental SVD, and subspace tracking. An important remark there is that ***expansion-based methods*** can be traced back to Bunch & Nielsen (1978) who proposed a method for incremental SVD.
- In Appendix G, as promised in Remark 5, we present a *dual* approach to continual matrix factorization, where the storage consumption shrinks over time.

B. Elaboration on The Main Paper

Here we take the opportunity to elaborate on several points omitted in the main paper due to the lack of space.

B.1. Elaboration on Terminologies

- At first glance the Ideal Continual Learner seems to implicitly assume that the task identities are given during training, and therefore it can not be applied to the more general and more challenging case of *task-agnostic* or *task-free* continual learning, where in the training phase the learner has no access to task identities (Zeno et al., 2018; 2021; Aljundi et al., 2019a; Lee et al., 2020; Jin et al., 2021; Wang et al., 2022c; Pourcel et al., 2022; Ye & Bors, 2022). However, for task-agnostic or task-free continual learning, we receive a batch of samples each time, and we can simply regard it as a task, and compute the set of common global minimizers for this batch of data. Then we can formulate (2) for the next task (i.e., the next batch). Summarized informally, `ICL` is task-agnostic. However, a major limitation of this task-agnostic formulation of `ICL` is that it leads to a very large T, which compromises the generalization bounds derived in the main paper.
- Methods of Remark 3 project the gradients onto certain subspaces, so they can be thought of as regularizing the gradients. This is why these methods are called ***regularization-based methods*** in the taxonomy of Qu et al. (2021). In the paper, we call them ***memory-based optimization methods***, as they need to store some projection matrices and we believe it is important to highlight this extra memory consumption for continual learning. We used the phrase ***expansion-based methods***, while ***architecture-based methods*** and ***structure-based methods*** are popular alternatives.
- While our formulation of the continual learning problem is general (e.g., each task can have different losses), our examples of continual linear regression and continual matrix factorization (§3) are specific in the sense that each task has the same loss and it is just that data samples are different for every task. This specific setting is called *domain-incremental learning* in the taxonomy of van de Ven et al. (2022). Quote van de Ven et al. (2022):

  > *Using task-specific components in this scenario is, however, only possible if an algorithm first identifies the task, but that is not necessarily the most efficient strategy. Preventing forgetting 'by design' is therefore not possible with domain-incremental learning, and alleviating catastrophic forgetting is still an important unsolved challenge.*

  That said, we proved that continual learning without forgetting is possible under Assumption 1.

B.2. Elaboration on Memory Selection Methods

Here we review related works on memory selection methods for rehearsal in light of Theorem 3.

Arguably, rehearsal is a very simple and effective idea that balances memory consumption and the amount of forgetting. It has been commonly believed that, quoting Chaudhry et al. (2019), "*... the sample that the learner selects to populate the memory becomes crucial*". Based on prior works (Vitter, 1985; Lopez-Paz & Ranzato, 2017; Rebuffi et al., 2017; Riemer et al., 2019), Chaudhry et al. (2019) summarized four (seven resp.) basic methods for selecting samples. More recently, based on Isele & Cosgun (2018); Hayes & Kanan (2021), Araujo et al. (2022) further included three more basic methods, and compared the seven methods for natural language processing applications. The setting of Araujo et al. (2022) is domain-incremental learning, and our generalization bound (19) applies to the rehearsal mechanism verbatim, under the same assumptions. (Our result for the constrained optimization formulation is not applicable as it requires task identities.)

Table 1 of Araujo et al. (2022) presents the performance of these seven methods for text classification and question answering, while Table 2 presents running times. From the two tables, one can observe the performance difference between **N. Random** and **Reservoir** is statistically insignificant. Figure 1 of Araujo et al. (2022) further shows that the two sampling methods, **N. Random** and **Reservoir**, tend to keep nearly the same number of samples for each task, while the samples that **N. Random** selects and those that **Reservoir** selects are in general very different. This corresponds well to Theorem 3, which formalizes an intuitive fact that selecting which samples to store does not matter under if the samples within each task fulfill the i.i.d. assumption. In particular, our generalization bound (19) is sensitive mainly to the number of samples for every task.

It is thus important to ruminate on whether selecting the so-called *representative* or *diverse* samples is relevant for improving continual learning systems (Araujo et al., 2022). We believe that selecting representative samples from each task or in a streaming setting can still be beneficial (Borsos et al., 2020; Sun et al., 2022) if the datasets contain some out-of-distribution data. This prompts incorporating an out-of-distribution detection module into continual learning frameworks.

C. Orthogonal Gradient Descent and Proposition 6

C.1. Orthogonal Gradient Descent for Deep Continual Learning

We first review the orthogonal gradient descent algorithm (OGD) of Farajtabar et al. (2020) for deep continual learning in image classification. In this image classification application, the dataset D_t for task t is $\{\boldsymbol{x}_{ti}, \boldsymbol{y}_{ti}\}_{i=1}^{m_t}$, where $\boldsymbol{x}_{ti}$ is some input image and $\boldsymbol{y}_{ti}$ is an one-hot vector of class labels. The training objective for task t in this context is typically written as

$$\min_{\boldsymbol{w}\in\mathbb{R}^n} \sum_{i=1}^{m_t} \ell_{\text{CE}}(f(\boldsymbol{w}; \boldsymbol{x}_{ti}); \boldsymbol{y}_{ti}),$$

where $\ell_{\text{CE}}(\cdot,\cdot)$ is the softmax cross entropy loss, and f represents a deep network parameterized by $\boldsymbol{w} \in \mathbb{R}^n$. The key idea of OGD is that, after training on task 1 via gradient descent and obtaining a good (if not optimal) estimate $\widetilde{\boldsymbol{w}}_1$, the training on task 2 is still via gradient descent, but the gradient is replaced by its projection onto the orthogonal complement of the range space of $\nabla f_1 := [\nabla f(\widetilde{\boldsymbol{w}}_1; \boldsymbol{x}_{11}), \ldots, \nabla f(\widetilde{\boldsymbol{w}}_1; \boldsymbol{x}_{1m_1})]$. Here, the gradient ∇ is evaluated with respect to the first parameter of f (e.g., network weights). This idea is easily extended to multiple tasks. In particular, suppose we are now going to solve task t. For $i = 1, \ldots, t-1$, let $\widetilde{\boldsymbol{w}}_i$ be the network weights obtained via training for tasks $1, \ldots, i$ sequentially. For training on task t, The OGD method performs gradient descent, with the gradient replaced by its projection onto the orthogonal complement of the range space of

$$\nabla F_{t-1} := [\nabla f_1, \ldots, \nabla f_{t-1}], \qquad \nabla f_i := [\nabla f(\widetilde{\boldsymbol{w}}_i; \boldsymbol{x}_{i1}), \ldots, \nabla f(\widetilde{\boldsymbol{w}}_i; \boldsymbol{x}_{im_i})] \ \ (\forall i = 1, \ldots, t-1).$$

In deep learning, the numbers of parameters and data samples are very large, so storing ∇F_{t-1} is prohibitive. This is improved by several recent follow-up works, which we mentioned in Remark 3 and will review in Appendix D.1. Note that we will derive a theoretical analysis of OGD for continual linear regression (Appendix C.3), and our analysis is also applicable to methods of Remark 3 under similar assumptions.

C.2. Orthogonal Gradient Descent for Continual Linear Regression

We now derive the OGD algorithm for continual linear regression. For each task t, the model is $f(\boldsymbol{w}; \boldsymbol{X}_t) := \boldsymbol{X}_t \boldsymbol{w}$ and the loss is the MSE loss. So we have $\nabla f_i = \boldsymbol{X}_i^\top$ for every $i = 1, \ldots, t-1$ (i.e., the gradient is constant). Therefore,

$$\nabla F_{t-1} := [\nabla f_1, \ldots, \nabla f_{t-1}] = [\boldsymbol{X}_1^\top, \ldots, \boldsymbol{X}_{t-1}^\top].$$

Denote by range$(\cdot)$ and null$(\cdot)$ the range space and nullspace of some matrix, respectively. By basic linear algebra we have

$$\begin{aligned}\big(\text{range}(\nabla F_{t-1})\big)^{\perp} &= \Big(\text{range}\big([\boldsymbol{X}_1^\top, \dots, \boldsymbol{X}_{t-1}^\top]\big)\Big)^{\perp} \\ &= \text{null}\big([\boldsymbol{X}_1^\top, \dots, \boldsymbol{X}_{t-1}^\top]^\top\big) \\ &= \text{null}(\boldsymbol{X}_1) \cap \dots \cap \text{null}(\boldsymbol{X}_{t-1}) \\ &= \text{range}(\boldsymbol{K}_{t-1}).\end{aligned} \tag{20}$$

Here we recall that we defined $\boldsymbol{K}_{t-1}$ as an orthonormal basis matrix of null$(\boldsymbol{X}_1) \cap \dots \cap$ null$(\boldsymbol{X}_{t-1})$ in §3.1. Since $\boldsymbol{K}_{t-1}^\top \boldsymbol{K}_{t-1}$ is the identity matrix, the matrix representation of this projection is $\boldsymbol{K}_{t-1}\boldsymbol{K}_{t-1}^\top$. When solving task t, OGD projects the gradient of the loss $\|\boldsymbol{X}_t \boldsymbol{w} - \boldsymbol{y}_t\|_2^2$ at each iteration onto range$(\boldsymbol{K}_{t-1})$, and then performs a descent step with this modified gradient. We next describe the OGD algorithm more formally.

Let k be an iteration counter, $\gamma_t^{(k)}$ the stepsize for task t at iteration k. Set the initialization $\boldsymbol{w}_t^{(0)}$ for task t as the final weight $\widetilde{\boldsymbol{w}}_{t-1}$ after training the previous tasks sequentially, i.e., $\boldsymbol{w}_t^{(0)} \leftarrow \widetilde{\boldsymbol{w}}_{t-1}$, while for the first task we use an arbitrary initialization $\boldsymbol{w}_1^{(0)}$. For task 1, OGD performs gradient descent

$$\boldsymbol{w}_1^{(k+1)} \leftarrow \boldsymbol{w}_1^{(k)} - \gamma_1^{(k)} \boldsymbol{h}_t^{(k)}, \quad \boldsymbol{h}_1^{(k)} := 2\boldsymbol{X}_1^\top(\boldsymbol{X}_1 \boldsymbol{w}_1^{(k)} - \boldsymbol{y}_1).$$

For task $t > 1$, its update rule is defined as (recall (9))

$$\boldsymbol{w}_t^{(k+1)} \leftarrow \boldsymbol{w}_t^{(k)} - \gamma_t^{(k)} \cdot \boldsymbol{K}_{t-1}\boldsymbol{K}_{t-1}^\top \boldsymbol{h}_t^{(k)}, \quad \boldsymbol{h}_t^{(k)} := 2\boldsymbol{X}_t^\top(\boldsymbol{X}_t \boldsymbol{w}_t^{(k)} - \boldsymbol{y}_t). \tag{21}$$

C.3. Theoretical Analysis of Orthogonal Gradient Descent for Continual Linear Regression

In this section we interpret what we meant by Proposition 6. Let us start with a simplified situation where OGD is assumed to already find a common global minimizer of the first $t-1$ tasks.

Proposition 8. *Let $t > 1$. Suppose Assumption 1 holds for continual linear regression. Assume that the weight $\widetilde{\boldsymbol{w}}_{t-1}$ produced by OGD after training the first $t-1$ tasks is a common global minimizer of tasks $1, \dots, t-1$. Then the update formula (21) of OGD for task t is equivalent to a gradient descent step applied to the objective (8) of* `ICL`†.

Proof. Since OGD is initialized at a common global minimizer $\boldsymbol{w}_t^{(0)} = \widetilde{\boldsymbol{w}}_{t-1}$, we can write $\boldsymbol{w}_t^{(0)} = \widetilde{\boldsymbol{w}}_{t-1} + \boldsymbol{K}_{t-1}\boldsymbol{a}_t^{(0)}$ for a unique $\boldsymbol{a}_t^{(0)}$ (actually $\boldsymbol{a}_t^{(0)} = 0$). Define $\overline{\boldsymbol{X}}_t := \boldsymbol{X}_t \boldsymbol{K}_{t-1}$ and

$$\boldsymbol{a}_t^{(1)} := \boldsymbol{a}_t^{(0)} - \gamma_1^{(0)} \boldsymbol{g}_1^{(0)}, \quad \boldsymbol{g}_1^{(0)} := 2\overline{\boldsymbol{X}}_t^\top(\overline{\boldsymbol{X}}_t \boldsymbol{a}_t^{(0)} + \boldsymbol{X}_t \widetilde{\boldsymbol{w}}_{t-1} - \boldsymbol{y}_t). \tag{22}$$

Note that (22) is exactly a gradient descent step applied to (8) (with $\hat{\boldsymbol{w}}_{t-1}$ replaced by a "different" particular solution $\widetilde{\boldsymbol{w}}_{t-1}$ of first $t-1$ tasks). As per (21), the first iteration of OGD is

$$\begin{aligned} &\boldsymbol{w}_t^{(1)} \leftarrow \boldsymbol{w}_t^{(0)} - \gamma_t^{(0)} \cdot \boldsymbol{K}_{t-1}\boldsymbol{K}_{t-1}^\top\big(2\boldsymbol{X}_t^\top(\boldsymbol{X}_t \boldsymbol{w}_t^{(0)} - \boldsymbol{y}_t)\big) \\ \Leftrightarrow\ &\boldsymbol{w}_t^{(1)} = \widetilde{\boldsymbol{w}}_{t-1} + \boldsymbol{K}_{t-1}\boldsymbol{a}_t^{(0)} - \gamma_t^{(0)} \cdot \boldsymbol{K}_{t-1}\Big(2\overline{\boldsymbol{X}}_t^\top\big(\boldsymbol{X}_t(\widetilde{\boldsymbol{w}}_{t-1} + \boldsymbol{K}_{t-1}\boldsymbol{a}_t^{(0)}) - \boldsymbol{y}_t\big)\Big) \\ \Leftrightarrow\ &\boldsymbol{w}_t^{(1)} = \widetilde{\boldsymbol{w}}_{t-1} + \boldsymbol{K}_{t-1}(\boldsymbol{a}_t^{(0)} - \gamma_t^{(0)} \boldsymbol{g}_1^{(0)}) \\ \Leftrightarrow\ &\boldsymbol{w}_t^{(1)} = \widetilde{\boldsymbol{w}}_{t-1} + \boldsymbol{K}_{t-1}\boldsymbol{a}_t^{(1)} \end{aligned} \tag{23}$$

Now we see from (22) and (23) that the first iteration of OGD can be thought of first computing the coefficient vector $\boldsymbol{a}_t^{(1)}$ via gradient descent (22) and then updating $\boldsymbol{w}_t^{(1)}$ via $\boldsymbol{w}_t^{(1)} = \widetilde{\boldsymbol{w}}_{t-1} + \boldsymbol{K}_{t-1}\boldsymbol{a}_t^{(1)}$. Since $\widetilde{\boldsymbol{w}}_{t-1}$ and $\boldsymbol{K}_{t-1}$ are fixed, the update of $\boldsymbol{w}_t^{(1)}$ in the first iteration of OGD can actually be viewed as calculating $\boldsymbol{a}_t^{(1)}$ implicitly. Finally, one can easily verify that the above derivation applies to any iteration k (e.g., by changing the indices from 0 to k and from 1 to $k+1$). □

With a suitable choice of stepsizes, gradient descent converges to an optimal solution to (8) at the number of iterations k approaches infinity; see, e.g., Theorem 2.1.14 of Nesterov (2018). As a consequence, under the assumption of Proposition 8, the OGD algorithm applied to continual linear regression converges to an optimal solution of (7) as $k \to \infty$.

The above analysis relies on simplified assumptions that OGD can find an exact solution to the first $t-1$ tasks, and that one can run OGD for infinitely many iterations. Dispensing with these assumptions to reach a similar conclusion is not hard:

Proposition 9. *Assume $t > 1$. Suppose Assumption 1 holds for continual linear regression. Let $\widetilde{\boldsymbol{w}}_{t-1}$ be the weight produced by the OGD algorithm after training the first $t-1$ tasks. Then the update formula (21) of the OGD algorithm for task t is equivalent to a gradient descent step applied to the following problem*

$$\min_{\boldsymbol{a}} \|\boldsymbol{X}_t(\widetilde{\boldsymbol{w}}_{t-1} + \boldsymbol{K}_{t-1}\boldsymbol{a}) - \boldsymbol{y}_t\|_2^2. \tag{24}$$

Proof. The proof follows directly from that of Proposition 8. □

The only difference of (24) from the objective (8) of $\texttt{ICL}^\dagger$ is that $\widetilde{\boldsymbol{w}}_{t-1}$ now might not be optimal for all previous tasks. Therefore, OGD is approximately equivalent to $\texttt{ICL}^\dagger$, and its approximation quality depends on how close $\widetilde{\boldsymbol{w}}_{t-1}$ is to the set $\mathcal{K}_{t-1}$ of common solutions to previous $t-1$ tasks.

D. Continual Learning: Related Works

There are now many existing methods for (deep) continual learning, including ***regularization-based methods*** (Kirkpatrick et al., 2017; Zenke et al., 2017; Liu et al., 2018; Ritter et al., 2018; Lesort et al., 2019; Park et al., 2019; Yin et al., 2020; Zhou et al., 2021; Heckel, 2022), ***memory-based methods*** (Robins, 1993; Shin et al., 2017; Rebuffi et al., 2017; Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2019; Aljundi et al., 2019b; Buzzega et al., 2020; Verwimp et al., 2021; Bang et al., 2021; Kim et al., 2022; Zhang et al., 2022; Wang et al., 2022a; Saha & Roy, 2023), ***expansion-based methods*** (Rusu et al., 2016; Yoon et al., 2018; Mallya & Lazebnik, 2018; Li et al., 2019; Hung et al., 2019; Ramesh & Chaudhari, 2022; Douillard et al., 2022), or combinations thereof (Yan et al., 2021; Xie et al., 2022; Wang et al., 2022b; Frascaroli et al., 2023); for surveys, see, e.g., Parisi et al. (2019); De Lange et al. (2022); Qu et al. (2021); Shaheen et al. (2022); van de Ven et al. (2022).

Our review here has two purposes. First, we will discuss the methods of Remark 3 in detail as they are highly related to $\texttt{ICL}^\dagger$. Second, we will discuss related theoretical works through the lens of task relationships.

D.1. Memory-Based Optimization Methods (Projection-Based Methods)

The Basic Idea. Suppose now we have a three-layer deep network f in the following form:

$$f\big((\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3); \boldsymbol{X}_t^{(0)}\big) := \sigma_3(\sigma_2(\sigma_1(\boldsymbol{X}_t^{(0)}\boldsymbol{W}_1)\boldsymbol{W}_2)\boldsymbol{W}_3) \tag{25}$$

Here, each σ_i is some non-linearity activation function, $\boldsymbol{W}_i$ matrices of learnable weights, and $\boldsymbol{X}_t^{(0)}$ an input data matrix at the t-th batch (or for the t-th task). Every column of $\boldsymbol{X}_t^{(0)}$ is a data sample, and the number of columns corresponds to the batch size. We consider three layers as an example; what follows applies to multiple layers and is without loss of generality.

Let us write the network f (25) equivalently as

$$f\big((\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3); \boldsymbol{X}_t^{(0)}\big) = \sigma_3(\boldsymbol{X}_t^{(2)}\boldsymbol{W}_3), \quad \boldsymbol{X}_t^{(2)} := \sigma_2(\boldsymbol{X}_t^{(1)}\boldsymbol{W}_2), \quad \boldsymbol{X}_t^{(1)} := \sigma_1(\boldsymbol{X}_t^{(0)}\boldsymbol{W}_1). \tag{26}$$

To proceed, suppose for every $j = 0, 1, 2$ the intersection $\cap_{i=1}^{t-1}\text{null}(\boldsymbol{X}_i^{(j)})$ of nullspaces is not empty, and denote by $\boldsymbol{K}_{t-1}^{(j)}$ an orthonormal basis matrix of $\cap_{i=1}^{t-1}\text{null}(\boldsymbol{X}_i^{(j)})$. This matrix $\boldsymbol{K}_{t-1}^{(j)}$ plays the same role as the matrix $\boldsymbol{K}_{t-1}$ that we defined for continual linear regression (§3.1), and the difference is that we now defined such matrices for every linear layer.

Suppose now we have the data $\boldsymbol{X}_t^{(0)}$ for task (batch) t available. Let $\boldsymbol{G}_1, \boldsymbol{G}_2$ and $\boldsymbol{G}_3$ be the gradients calculated via data $\boldsymbol{X}_t^{(0)}$ and they account respectively for weights $\boldsymbol{W}_1, \boldsymbol{W}_2$, and $\boldsymbol{W}_3$. Let $\gamma_1, \gamma_2, \gamma_3$ be stepsizes. Then the weight update

$$\begin{aligned}
\boldsymbol{W}_1^+ &\leftarrow \boldsymbol{W}_1 - \gamma_1 \boldsymbol{K}_{t-1}^{(1)}(\boldsymbol{K}_{t-1}^{(1)})^\top \boldsymbol{G}_1 \\
\boldsymbol{W}_2^+ &\leftarrow \boldsymbol{W}_2 - \gamma_2 \boldsymbol{K}_{t-1}^{(2)}(\boldsymbol{K}_{t-1}^{(2)})^\top \boldsymbol{G}_2 \\
\boldsymbol{W}_3^+ &\leftarrow \boldsymbol{W}_3 - \gamma_2 \boldsymbol{K}_{t-1}^{(3)}(\boldsymbol{K}_{t-1}^{(3)})^\top \boldsymbol{G}_3
\end{aligned} \tag{27}$$

never changes the output of f on previous batches; for every $i < t$, it holds that

$$\begin{aligned}
&\boldsymbol{X}_i^{(0)}\boldsymbol{W}_1^+ = \boldsymbol{X}_i^{(0)}\boldsymbol{W}_1, \quad \boldsymbol{X}_i^{(1)}\boldsymbol{W}_2^+ = \boldsymbol{X}_i^{(1)}\boldsymbol{W}_2, \quad \boldsymbol{X}_i^{(2)}\boldsymbol{W}_3^+ = \boldsymbol{X}_i^{(2)}\boldsymbol{W}_3 \\
\Rightarrow\ & f\big((\boldsymbol{W}_1^+, \boldsymbol{W}_2^+, \boldsymbol{W}_3^+); \boldsymbol{X}_i^{(0)}\big) = f\big((\boldsymbol{W}_1, \boldsymbol{W}_2, \boldsymbol{W}_3); \boldsymbol{X}_i^{(0)}\big).
\end{aligned}$$

Yet, (27) can decrease the loss of the current data $\boldsymbol{X}_t^{(0)}$. This makes storing $\boldsymbol{K}_{t-1}^{(j)}$ an appealing approach for preventing forgetting. In reality, each $\boldsymbol{X}_i^{(j)}$ might not have a nullspace (for a small enough batch size), and the intersection $\cap_{i=1}^{t-1}\text{null}(\boldsymbol{X}_i^{(j)})$ might simply be empty. A simple and effective rescue is to perform certain low-rank approximations; we will not go into the details here, and we omit some practically important implementation details.

The Literature. The literature of continual learning has followed a slightly longer and perhaps more vivid trajectory to eventually identify the role of the projection matrix $\boldsymbol{K}_{t-1}^{(j)}(\boldsymbol{K}_{t-1}^{(j)})^\top$ in resisting forgetting. We review this trajectory next.

Inspired by classic works on adaptive filtering (Haykin, 2002) and recursive least-squares (Singhal & Wu, 1989) in signal processing, Zeng et al. (2019) proposed to set the projection matrix to be $I - (\boldsymbol{X}_1^{(j)})^\top\big(\boldsymbol{X}_1^{(j)}(\boldsymbol{X}_1^{(j)})^\top + \lambda I\big)^{-1}\boldsymbol{X}_1^{(j)}$ for the task 1 and update it for later tasks as in recursive least-squares. Here $\lambda > 0$ is some hyper-parameter and I identity matrix, and they are used to prevent the case where $\boldsymbol{X}_1^{(j)}(\boldsymbol{X}_1^{(j)})^\top$ is not invertible. The method of Zeng et al. (2019) can be understood as *recursive least-squares applied to deep networks in a layer-wise manner*.

If the batch size is large enough, then $\boldsymbol{X}_1^{(j)}(\boldsymbol{X}_1^{(j)})^\top$ is in general invertible, and if $\lambda = 0$ then $I - (\boldsymbol{X}_1^{(j)})^\top\big(\boldsymbol{X}_1^{(j)}(\boldsymbol{X}_1^{(j)})^\top\big)^{-1}\boldsymbol{X}_1^{(j)}$ is exactly a projection onto the nullspace of $\boldsymbol{X}_1^{(j)}$. While Guo et al. (2022) showed that updating λ in a certain way at every batch improves the empirical performance, inverting $\boldsymbol{X}_1^{(j)}(\boldsymbol{X}_1^{(j)})^\top + \lambda I$ might not be necessary if the goal is to compute (approximate) the nullspace of $\boldsymbol{X}_1^{(j)}$, which can be done via SVD. In fact, doing SVD would also eliminate the hyper-parameter λ.

Improvements over the strategy of Zeng et al. (2019) are given by independent efforts from Wang et al. (2021b) and Saha et al. (2021). Wang et al. (2021b) maintain a *weighted*[4] sum of covariance matrix $(\boldsymbol{X}_i^{(j)})^\top\boldsymbol{X}_i^{(j)}$, where i ranges over previous tasks $1, \dots, t-1$, and then extract an orthonormal basis matrix $K^{(j)}$ the nullspace of some low-rank approximation of this weighted sum. Such $K^{(j)}$ would be the desired $\boldsymbol{K}_{t-1}^{(j)}$ defined above, if the intersection $\cap_{i=1}^{t-1}\text{null}(\boldsymbol{X}_i^{(j)})$ were not empty. On the other hand, Saha et al. (2021) maintain an orthonormal basis for a low-rank approximation of $[(\boldsymbol{X}_1^{(j)})^\top, \dots, (\boldsymbol{X}_{t-1}^{(j)})^\top]$, and they project the gradient onto the orthogonal complement of the subspace spanned by this basis. In light of (20), the methods of Saha et al. (2021) and Wang et al. (2021b) are in principle equivalent, and their difference in performance (as reported by Kong et al. (2022)) relies on implementation details. Furthermore, their approaches are dual to each other and are the *two sides of the same coin*: Wang et al. (2021b) (approximately) maintains a basis for $\cap_{i=1}^{t-1}\text{null}(\boldsymbol{X}_i^{(j)})$ and Saha et al. (2021) for the orthogonal complement of $\cap_{i=1}^{t-1}\text{null}(\boldsymbol{X}_i^{(j)})$; see Appendix G for another example of this duality.

We stress that the provable success of these methods relies on Assumption 1, or otherwise pathological cases might arise; recall Example 1 (Past = Present) and see Lin et al. (2022a) for a different example. Finally, we refer the reader to Kong et al. (2022); Liu & Liu (2022); Lin et al. (2022a) for further developments on this line of research.

D.2. Theoretical Aspects of Continual Learning Through The Lens of Task Relationship

In order to develop any useful theory for continual learning or related problems that involve multiple tasks, the first step is to make suitable assumptions on the *task relationship* (or *task similarity*, or *task relatedness*). In the literature, there are several candidate assumptions available for modeling the task relationship, which we review here.

One early assumption on the task relationship is given by Baxter (2000) in the context of *learning to learn*, who assumed a statistical data and task generation model such that, a hypothesis space that contains good solutions for a new task can be learned from sufficiently many samples of old tasks. An analogous example can be made in our context: $\texttt{ICL}^\dagger$ learns $\mathcal{K}_t$ from old tasks, which serves as the hypothesis space for the next task.

The task relationship proposed by Ben-David & Borbely (2008) involves a family of functions $\mathcal{F}$. Ben-David & Borbely (2008) define two tasks as $\mathcal{F}$*-related* if the data generation distribution of one task can be transformed into that of the other by some function in $\mathcal{F}$. This assumption is adopted recently by Ramesh & Chaudhari (2022) and Prado & Riddle (2022) to provide insights on generalization for continual learning. A drawback of this assumption is that the $\mathcal{F}$-relatedness is typically characterized by an abstract and combinatorial notion of the *generalized VC dimension*, which in our opinion is less intuitive and less "tangible" than Assumptions 1 and 2.

Lee et al. (2021) and Asanuma et al. (2021) contextualize the task relationship in a teacher-student network setup. The tasks

[4] In principle (say if the nullspaces of $\boldsymbol{X}_i^{(j)}$ intersect), different weights would result in the same nullspace.

are defined by the teacher networks and the similarity of two given tasks is quantified by the difference in the weights of the two corresponding teacher networks. It appears to us that the insights Lee et al. (2021) and Asanuma et al. (2021) delivered under this teacher-student assumption are now limited to only two tasks.

Yet another task relationship is that every objective L_t is the composition of two functions $f_t \circ g$, where the function $g : \mathbb{R}^n \to \mathbb{R}^s$ is referred to as the *shared representation*; see, e.g., Maurer et al. (2016); Balcan et al. (2015); Tripuraneni et al. (2021); Du et al. (2021). The point is that, if g has been learned from old tasks and fixed, and if $n \gg s$, learning a new task would be more sample-efficient as the dimension of the search space reduces from n to a much smaller s. Li et al. (2022) and Cao et al. (2022) recently explored this idea in the continual learning context with certain new insights, while their settings are from ours. It is possible to obtain stronger generalization bounds and improve our results by leveraging this assumption of a shared representation, at the cost of obtaining a less general theorem than Theorems 1 and 2.

The task relationship that we consider in this paper is defined by Assumption 1 or 2, that is all tasks share at least a common global minimizer. As mentioned in the main paper, this assumption generalizes those of Evron et al. (2022) and Peng & Risteski (2022). While we show that Assumptions 1 and 2 are quite powerful and leads to many pleasant results, in the future we would like to study how strong Assumption 1 or 2 is for continual learning and come up with other alternatives.

E. Set-Theoretical Estimation

In the main paper, we mentioned that $\texttt{ICL}^\dagger$ is related to *set-theoretical estimation* in control. Here, we elaborate on this statement by reviewing related works and interpreting $\texttt{ICL}^\dagger$ from a control and set-theoretical perspective.

E.1. Ideal Continual Learner† = Predictor + Corrector

State estimation is a fundamental problem in control. From the data $D_0, D_1, \ldots, D_T$ observed sequentially over time and the initial state $\boldsymbol{w}_0 \in \mathcal{W} \subset \mathbb{R}^n$, the goal of state estimation is to estimate the subsequent states $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_T \in \mathbb{R}^n$, assuming the following equations[5]:

$$\begin{aligned} \boldsymbol{w}_t &= f_{t-1}(\boldsymbol{w}_{t-1}) \\ D_{t-1} &= h_{t-1}(\boldsymbol{w}_{t-1}) \end{aligned} \qquad t = 1, \ldots, T \tag{28}$$

Here, f_t is some known function that transfer state $\boldsymbol{w}_t$ into $\boldsymbol{w}_{t+1}$, and h_t is a known function that maps state $\boldsymbol{w}_t$ into our observation D_t. Of course, T is allowed to be ∞.

Since the initial state $\boldsymbol{w}_0$ belongs to $\mathcal{W}$ and the subsequent state $\boldsymbol{w}_t$ fulfills (28), $\boldsymbol{w}_t$ must belong to some set depending on $\mathcal{W}$, data $D_0, \ldots, D_T$, and (28). Such dependency was described by Witsenhausen (1968) in the case where f_{t-1} and h_{t-1} are linear maps, and was made mathematically precise by Kieffer et al. (1998) with a recursive algorithm. Defining $K_0 \leftarrow \mathcal{W}$, the algorithm of Kieffer et al. (1998) consists of two steps for every $t = 1, \ldots, T$:

$$\begin{aligned} &\text{(Prediction)} \quad G_t \leftarrow f_{t-1}(K_{t-1}) \\ &\text{(Correction)} \quad K_t \leftarrow h_t^{-1}(D_t) \cap G_t \end{aligned}$$

In words, the prediction step collects *all possible* states G_t that can be transferred from K_{t-1}, and the correction step rules out the states that are inconsistent with data D_t.

In light of Kieffer et al. (1998) and as the notations suggest, we have the following description of $\texttt{ICL}^\dagger$:

Proposition 10 (Ideal Continual Learner† = Predictor + Corrector)**.** *Under Assumption 1, the recursion (2) of $\texttt{ICL}^\dagger$ is equivalent to*

$$\begin{aligned} &\text{(Prediction)} \quad \mathcal{G}_t \leftarrow \operatorname*{argmin}_{w \in \mathcal{W}} L_t(\boldsymbol{w}; D_t) \\ &\text{(Correction)} \quad \mathcal{K}_t \leftarrow \mathcal{K}_{t-1} \cap \mathcal{G}_t \end{aligned} \tag{29}$$

The prediction step (29) involves solving a single task, while the correction step requires computing the intersection of two complicated sets. Finally, the prediction-correction description also requires storing the knowledge representation $\mathcal{K}_t$.

[5]This is a simplified model, e.g., we did not describe the *control signals* that enable state transfer, but this is enough for illustration.

E.2. Ideal Continual Learner† = Set Intersection Learner†?

Background. Combettes (1993) formulated a general estimation problem that encompasses many signal processing and control applications. The formulation of Combettes (1993) can be described as follows. Suppose we are given some information $D_1, \ldots, D_T$, from which we want to estimate a certain variable $\hat{\boldsymbol{w}}$. Each piece of information D_t determines a set $\mathcal{G}_t$ in which $\boldsymbol{w}^*$ lies. Assuming we can compute G_t from D_t, the problem of estimating $\hat{\boldsymbol{w}}$ reduces to computing (a point of) the set intersection $\cap_{t=1}^T G_t$. If the information we have is *inconsistent* (e.g., due to noise or out-of-distribution measurements), the intersection $\cap_{t=1}^T G_t$ might be empty. This is a more challenging situation, which might result in an NP-hard problem (as we discussed in the main paper). As in Assumption 1, we assume $\cap_{t=1}^T G_t$ is non-empty, therefore the difficulty of finding a point in $\cap_{t=1}^T G_t$ largely depends on the *shape* of each set G_t.

Computing Set Intersections. Suppose $\cap_{t=1}^T G_t \neq \varnothing$. Finding a point in $\cap_{t=1}^T G_t$ is a classic problem and has been of great interest nowadays due to many modern applications. Classic methods include the alternating projection method of John von Neumann in 1933 for the case of two intersecting affine subspaces (as Lindstrom & Sims (2021) surveyed); the methods of Karczmarz (1937) and Cimmino (1938) for the case where G_t's are finitely many affine hyperplanes (as Combettes (1993) reviewed); the method of Boyle & Dykstra (1986) for computing a point in the intersection of two convex sets; as well as many extensions of these methods for finitely many convex sets and even non-convex sets. We refer the reader to Borwein & Tam (2014) for some equivalence among these methods and to Theodoridis et al. (2010); Lindstrom & Sims (2021) and some references of Borwein & Tam (2014) for surveys. See also Evron et al. (2022) where some extensions of Karczmarz (1937) were discussed in a continual learning context.

The Continual Learning Context. The Ideal Continual Learner† (`ICL`†) might be viewed as a method that computes the set $\mathcal{G}_t$ of global minimizes for each task t and then computes their intersection $\cap_{t=1}^T \mathcal{G}_t$. If all objectives L_t's are convex, then in general $\mathcal{G}_t$ is a convex set, and it is possible to implement `ICL`† via computing intersections of convex sets. However, there are still several issues that make the continual learning problem harder than computing set intersections:

- The first difficulty is to find a representation of $\mathcal{G}_t$ that is amenable to alternating optimization (we discussed several equivalent representations of $\mathcal{G}_t$ in §2). Such representation should consume as less memory as possible, and yet might purely rely on a user's choice; e.g., one might revisit past tasks as an implicit way of storing $\mathcal{G}_t$ (Evron et al., 2022).
- The other difficulty occurs when there are infinitely many tasks ($T = \infty$), and this basically rules out the chance of storing all sets $\mathcal{G}_t$'s. Similar issues have been considered in the signal processing context, where one is given the set $\mathcal{G}_t$ sequentially; see Theodoridis et al. (2010) for a review. However, theoretical guarantees of existing methods in that line of research are about convergence to $\cap_{t=t_0}^\infty \mathcal{G}_t$ for some number t_0 (potentially unknown), not to $\cap_{t=1}^\infty \mathcal{G}_t$. The reason that t_0 is in general unknown is this: These methods are some variants of alternating projections, which might *forget* the very first set $\mathcal{G}_1$ after projecting onto later sets.

F. Streaming PCA, Incremental SVD, and Subspace Tracking

As mentioned in the main paper, the continual matrix factorization formulation is related to several classic problems, *streaming principal component analysis* (PCA), *incremental singular value decomposition* (SVD), and *subspace tracking*. These problems have been extensively studied in machine learning, numerical linear algebra, and signal processing communities. While we refer the reader to Balzano et al. (2018) and Vaswani et al. (2018) for comprehensive reviews, we discuss several related methods here, aiming at highlighting their connections to continual learning.

Streaming PCA. PCA refers to modeling a data matrix $[\boldsymbol{Y}_1 \ \cdots \ \boldsymbol{Y}_T]$ with some low-dimensional subspace, and it is a century-old topic (Pearson, 1901) that has many applications and extensions (Jolliffe, 2002; Vidal et al., 2016). Streaming PCA differs from PCA in that one receives $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_T$ sequentially, and each $\boldsymbol{Y}_t$ is seen only once. A popular method for streaming PCA is due to Oja (1982). Assume we want to project the data onto a r-dimensional subspace of $\mathbb{R}^n$, with r a hyper-parameter. The method of Oja (1982) initializes a random orthonormal matrix $\boldsymbol{U}^{(1)} \in \mathbb{R}^{n \times r}$, updates

$$\boldsymbol{U}^{(k+1)} \leftarrow \operatorname{orth}\Big(\boldsymbol{U}^{(k)} + \gamma^{(k)} \boldsymbol{Y}_k \boldsymbol{Y}_k^\top \boldsymbol{U}^{(k)}\Big) \tag{30}$$

at iteration k with stepsize $\gamma^{(k)}$. Here $\operatorname{orth}(\cdot)$ orthogonalizes an input matrix (e.g., via QR or SVD decomposition). The update (30) can be regarded as a stochastic projected gradient descent method; in particular, $\gamma^{(k)} \to \infty$, then (30) becomes

a stochastic version of the power method that computes r eigenvectors of $[\boldsymbol{Y}_1 \ \cdots \ \boldsymbol{Y}_T]$ corresponding to r maximum eigenvalues. Similarly to SGD or the algorithm of Evron et al. (2022), in the worse case, this stochastic method forgets if it does not revisit past data, and even if $[\boldsymbol{Y}_1 \ \cdots \ \boldsymbol{Y}_T]$ is exactly of rank r; note that we did not give a formal proof for this claim, but it can be derived by combining the minimality of $\mathcal{K}_t$ (§2.3) with problem-specific arguments.

Incremental SVD. The incremental SVD problem is to compute the SVD of $[\boldsymbol{Y}_1 \ \boldsymbol{Y}_2]$ from the SVD of a matrix $\boldsymbol{Y}_1 = \boldsymbol{U}_1\boldsymbol{\Sigma}_1\boldsymbol{V}_1^\top$. This is a different and slightly more difficult goal from continual matrix factorization, which aims to estimate (an orthonormal basis matrix of) a subspace sum. Of course, we could do incremental SVD for increasingly more data, and extract the desired basis matrix from the final SVD decomposition of $[\boldsymbol{Y}_1 \ \cdots \ \boldsymbol{Y}_t]$; in other words, incremental SVD algorithms can be used to solve the continual matrix factorization problem. The issue is that it takes more memory to store the SVD than storing only a basis matrix as in our approach, and potentially takes more time for computation.

To better understand the matter, we review a popular incremental SVD method, described in Section 3 of Brand (2002) (see also Bunch & Nielsen (1978)). With $Y_2 := (I - \boldsymbol{U}_1\boldsymbol{U}_1^\top)\boldsymbol{Y}_2$ and a QR-decomposition $Q_2R_2 = Y_2$ of Y_2, we have

$$\begin{aligned}[\boldsymbol{Y}_1 \ \ \boldsymbol{Y}_2] &= [\boldsymbol{U}_1\boldsymbol{\Sigma}_1\boldsymbol{V}_1^\top \ \ \boldsymbol{Y}_2] \\ &= [\boldsymbol{U}_1 \ \ Q_2]\begin{bmatrix}\boldsymbol{\Sigma}_1 & \boldsymbol{U}_1^\top\boldsymbol{Y}_2 \\ 0 & R_2\end{bmatrix}\begin{bmatrix}\boldsymbol{V}_1 & 0 \\ 0 & I\end{bmatrix}^\top\end{aligned}$$

Since $[\boldsymbol{U}_1 \ \ Q_2]$ and $\begin{bmatrix}\boldsymbol{V}_1 & 0 \\ 0 & I\end{bmatrix}$ have orthonormal columns, computing the SVD of $[\boldsymbol{Y}_1 \ \ \boldsymbol{Y}_2]$ reduces to computing the SVD $U_2\boldsymbol{\Sigma}_2V_2^\top$ of the middle matrix $\begin{bmatrix}\boldsymbol{\Sigma}_1 & \boldsymbol{U}_1^\top\boldsymbol{Y}_2 \\ 0 & R_2\end{bmatrix}$. Indeed, with SVD $U_2\boldsymbol{\Sigma}_2V_2^\top$, we can obtain the desired decomposition of $[\boldsymbol{Y}_1 \ \ \boldsymbol{Y}_2]$ as $\boldsymbol{U}_2\boldsymbol{\Sigma}_2\boldsymbol{V}_2^\top$, where $\boldsymbol{U}_2$ and $\boldsymbol{V}_2$ are defined as

$$\boldsymbol{U}_2 \leftarrow [\boldsymbol{U}_1 \ \ Q_2]U_2, \quad \boldsymbol{V}_2 = \boldsymbol{V}_1V_2. \tag{31}$$

Note that this approach can also be regarded as an ***expansion-based method***, and by design it never forgets when applied to continual matrix factorization. However, it requires an extra QR decomposition, and it requires an SVD on a larger matrix than in our approach (§3.2). A subtler drawback is that this method performs SVD and then multiplication (31) to obtain the orthonormal basis. This is potentially numerically unstable; as Brand (2006) later commented:

Over thousands or millions of updates, the multiplications may erode the orthogonality of $\boldsymbol{U}'$ [e.g., $\boldsymbol{U}_2$] through numerical error... [Brand (2006) described some ad-hoc remedy] ... It is an open question how often this is necessary to guarantee a certain overall level of numerical precision.

This numerical issue does not exist in our approach, where our multiplication is followed by an SVD; then from the SVD we directly extract the desired basis matrix, which is typically orthonormal up to machine accuracy given the numerical stability of SVD algorithms (preconditioned if necessary). The reason that we could do so, though, stems from the fact that we need only to compute an orthonormal basis of the subspace sum $\mathcal{S}_1 + \cdots + \mathcal{S}_t$ for the continual matrix factorization problem, which is arguably simpler than incrementally updating the SVD. As a final remark, our approach can be extended to update the U and V matrices for incremental SVD, at the potential cost that updating the singular values could be numerically unstable. Nevertheless, discussing these is beyond the scope of the paper, and hence we omit the details.

Subspace Tracking. Suppose now we receive data $\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_T$ sequentially, with $\mathcal{S}_t = \text{range}(\boldsymbol{Y}_t)$. In the problem of subspace tracking, we assume $\mathcal{S}_t$ changes slowly, as t increases, and the goal at task t or time t is to estimate $\mathcal{S}_t$. To do so, a classic idea (e.g., see Yang (1995; 1996)) is to define a *forgetting* factor β with $0 \ll \beta \leq 1$, and minimize the following objective in a streaming fashion:

$$\min_{\boldsymbol{U}} \sum_{t=1}^{T} \beta^{T-t}\|\boldsymbol{Y}_t - \boldsymbol{U}\boldsymbol{U}^\top\boldsymbol{Y}_t\|_{\text{F}}^2. \tag{32}$$

Here, the forgetting factor β discounts the previous samples, accounting for the fact that $\mathcal{S}_t$ is slowly changing; more generally, such factor β also appears in many classic reinforcement learning algorithms as a *discount rate* to discount the past rewards (Sutton & Barto, 2018). There are also works that make mathematically precise assumptions on *how*

$\mathcal{S}_t$ changes over time. For example, $\mathcal{S}_t$ could be slowly rotated, or keep static for a sufficiently long time until changing (Narayanamurthy & Vaswani, 2018). Intuitively, making how $\mathcal{S}_t$ changes precise give chances to design better algorithms, although we shall not review these approaches. The point here is that, this classic line of research furnished the very rudimentary idea of what is recently known as *selective forgetting* or *active forgetting* in the deep continual learning context (Aljundi et al., 2018; Wang et al., 2021a; Shibata et al., 2021; Liu et al., 2022). Of course, achieving selective or active forgetting in a principled way for deep continual learning is much harder than for subspace tracking.

G. A Dual Approach to Continual Principal Component Analysis

Background. In §3.2, we considered continual matrix factorization where task t is associated with the loss $L_t\big((\boldsymbol{U},\boldsymbol{C});\boldsymbol{Y}_t\big) = \|\boldsymbol{U}\boldsymbol{C}-\boldsymbol{Y}_t\|_{\mathrm{F}}^2$, and this loss is equivalent to a PCA objective $\|\boldsymbol{U}\boldsymbol{U}^\top\boldsymbol{Y}_t-\boldsymbol{Y}_t\|_{\mathrm{F}}^2$ in variable $\boldsymbol{U}$, assuming $\boldsymbol{U}^\top\boldsymbol{U}$ is the identity matrix. Since $\boldsymbol{U}$ gradually grows its columns to accommodate new tasks, this might eventually consume too much memory. As promised in Remark 5, we now address this issue, using the following key observation: If $\boldsymbol{U}$ and $\boldsymbol{B}$ are orthonormal basis matrices[2] of some linear subspace and its orthogonal complement, respectively, then $\boldsymbol{U}\boldsymbol{U}^\top+\boldsymbol{B}\boldsymbol{B}^\top$ is the identity matrix, and we have

$$\|\boldsymbol{U}\boldsymbol{U}^\top\boldsymbol{Y}_t-\boldsymbol{Y}_t\|_{\mathrm{F}}^2 = \|\boldsymbol{B}\boldsymbol{B}^\top\boldsymbol{Y}_t\|_{\mathrm{F}}^2 = \|\boldsymbol{Y}_t^\top\boldsymbol{B}\|_{\mathrm{F}}^2. \tag{33}$$

Note that relation (33) was discussed by Tsakiris & Vidal (2018) in a different context, in comparison to the algorithm of Lerman et al. (2015). In that line of research, $\boldsymbol{U}$ is typically referred to as the *primal representation* of the subspace $\mathcal{S}_t$ and $\boldsymbol{B}$ the dual representation. To conclude, we can use $\|\boldsymbol{Y}_t^\top\boldsymbol{B}\|_{\mathrm{F}}^2$ as the objective for task t, and the aim for task t is to find an orthonormal basis matrix for subspace $\mathcal{S}_t^\perp$, which is uniquely identified with $\mathcal{S}_t$. The goal of *continual principal component analysis* is to find the subspace intersection $\cap_{i=1}^t\mathcal{S}_i^\perp$ upon encountering task t ($\forall t$).

Implementing The Ideal Continual Learner†. We now describe $\texttt{ICL}^\dagger$ for solving the problem of continual PCA. Denote by B_t a basis matrix for $\cap_{i=1}^t\mathcal{S}_i^\perp$ with $B_t^\top B_t = I$, we have $\|\boldsymbol{Y}_i^\top B_t\|_{\mathrm{F}} = 0$ for every $i=1,\dots,t$. Therefore, for each task, we maintain and update B_t. To start with, set B_1 to be the matrix whose columns are all right singular vectors of $\boldsymbol{Y}_t^\top$ corresponding to its zero singular values. Suppose now $t>1$ and we are given B_{t-1} and $\boldsymbol{Y}_t$, and we wish to compute a B_t. Of course $\cap_{i=1}^t\mathcal{S}_i^\perp \subset \cap_{i=1}^{t-1}\mathcal{S}_i^\perp$, hence B_t has fewer or equal columns than B_{t-1} and each column of B_t can be represented as a linear combination of columns of B_{t-1}. Therefore, parameterize B_t as $B_t = B_{t-1}\overline{C}_t$, and we solve

$$\begin{aligned}
&\overline{C}_t \in \operatorname*{argmin}_{C_t} \|\boldsymbol{Y}_t^\top B_{t-1}C_t\|_{\mathrm{F}}^2 \ \text{ s.t. } \ (B_{t-1}C_t)^\top B_{t-1}C_t = I \\
\Leftrightarrow\ &\overline{C}_t \in \operatorname*{argmin}_{C_t} \|\boldsymbol{Y}_t^\top B_{t-1}C_t\|_{\mathrm{F}}^2 \ \text{ s.t. } \ C_t^\top C_t = I,
\end{aligned}$$

where the equivalence is due to $B_{t-1}^\top B_{t-1} = I$. As a result, we can set $\overline{C}_t$ to be the matrix whose columns are all right singular vectors of $\boldsymbol{Y}_t^\top B_{t-1}$ corresponding to its zero singular values. This concludes how $\texttt{ICL}^\dagger$ can be implemented for continual PCA.

Summary. We now see *two sides of the same coin*: We can implement $\texttt{ICL}^\dagger$ for continual matrix factorization (or continual PCA), using either the primal or dual representation. The primal representation (U_t) grows while the dual representation (B_t) shrinks. This is not to say that the dual representation is always better. The dual representation saves memory (and is thus preferred) if and only if $\dim(\sum_{i=1}^t\mathcal{S}_i)$ exceeds half of the ambient dimension. Hence, a better implementation for continual matrix factorization (or continual PCA) would switch the representation from primal to dual, as t increases such that $\dim(\sum_{i=1}^t\mathcal{S}_i)$ is greater than that threshold. Nevertheless, the total memory needed in both representations is bounded above by $n\times\dim(\sum_{i=1}^t\mathcal{S}_i)$. This seems to contradict the results of Knoblauch et al. (2020) and Chen et al. (2022), which suggest that continual learning without forgetting would require memory to grow linearly with the number T of tasks. The catch is that, besides other differences in settings, their results are for the worst-case scenario, while our continual matrix factorization or continual PCA setting is quite specific and does not belong to the worst case.

H. Proofs

Proof of Proposition 1. Clearly $\mathcal{K}_1 = \mathcal{G}_1$. Assume $\mathcal{K}_{t-1} = \cap_{i=1}^{t-1}\mathcal{G}_i$. Since L_t is minimized over $\mathcal{G}_t$ (1) and $\cap_{i=1}^t\mathcal{G}_i \neq \varnothing$, the set of global minimizers of (2) is $\cap_{i=1}^t\mathcal{G}_i$. By definition (2) we have $\mathcal{K}_t = \cap_{i=1}^t\mathcal{G}_i$. The proof is complete by induction. □

Proof of Proposition 4. Let $\hat{\boldsymbol{w}}_t \in \mathcal{K}_t = \cap_{i=1}^t \mathcal{G}_i$, then we have

$$\begin{aligned} \min_{w \in \mathcal{W}} \sum_{i=1}^t \alpha_i L_i(\boldsymbol{w}; D_i) &\geq \sum_{i=1}^t \alpha_i \min_{w \in \mathcal{W}} L_i(\boldsymbol{w}; D_i) \\ &= \sum_{i=1}^t \alpha_i L_i(\hat{\boldsymbol{w}}_t; D_i). \end{aligned} \tag{34}$$

Since $\hat{\boldsymbol{w}}_t$ is feasible to the multitask objective (5), the minimum value of the multitask objective is attained exactly when each objective L_i is minimized, meaning that $\mathcal{K}_t$ is the set of global minimizers of the multitask objective. □

Proof of Theorem 1. Assumption 1 and Proposition 1 imply $\hat{\boldsymbol{w}}$ is a common global minimizer of all empirical tasks (1). For $t = 1, \ldots, T$, let $\boldsymbol{w}_t^* \in \mathcal{G}_t^*$ be a global minimizer of learning task t (12). Applying a union bound to (14), we get

$$\begin{aligned} \mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\hat{\boldsymbol{w}}; \boldsymbol{d})] - L_t(\hat{\boldsymbol{w}}; D_t) &\leq \zeta(m_t, \delta'), \\ L_t(\boldsymbol{w}_t^*; D_t) - \mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\boldsymbol{w}_t^*; \boldsymbol{d})] &\leq \zeta(m_t, \delta'), \end{aligned} \tag{35}$$

for all $t = 1, \ldots, T$, with probability at least $1 - T\delta'$. Summing up the inequalities of (35) and noticing $L_t(\hat{\boldsymbol{w}}; D_t) \leq L_t(\boldsymbol{w}_t^*; D_t)$ and $c_t^* = \mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\boldsymbol{w}_t^*; \boldsymbol{d})]$, we then obtain

$$\mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\hat{\boldsymbol{w}}; \boldsymbol{d})] \leq c_t^* + 2\zeta(m_t, \delta') = c_t^* + \zeta(m_t, \delta').$$

The last equality is due to the big-O notation (14). With $\delta := \delta'/T$ we finish the proof. □

Proof of Proposition 5. Given $(\boldsymbol{X}_1, \boldsymbol{y}_1)$, we can compute $\boldsymbol{K}_1$ via an SVD on $\boldsymbol{X}_1$, e.g., set $\boldsymbol{K}_1$ to be the matrix whose columns are right singular vectors of $\boldsymbol{X}_1$ corresponding to its zero singular values. We can compute a particular solution $\hat{\boldsymbol{w}}_1 := \boldsymbol{X}_1^\dagger \boldsymbol{y}_1$ to the normal equations $\boldsymbol{X}_1^\top \boldsymbol{X}_1 \boldsymbol{w} = \boldsymbol{X}_1^\top \boldsymbol{y}_1$; here $\boldsymbol{X}_1^\dagger$ denotes the pseudoinverse of $\boldsymbol{X}_1$, and can be computed from SVD of $\boldsymbol{X}_1$.

For the case $t > 1$, suppose we are given $(\hat{\boldsymbol{w}}_{t-1}, \boldsymbol{K}_{t-1})$ and data $(\boldsymbol{X}_t, \boldsymbol{y}_t)$. Then, by the definitions of $(\hat{\boldsymbol{w}}_{t-1}, \boldsymbol{K}_{t-1})$ and $\mathcal{K}_{t-1}$, the equivalence between (7) and (8) is immediate.

The normal equations of (8) are given by ($\overline{\boldsymbol{X}}_t := \boldsymbol{X}_t \boldsymbol{K}_{t-1}$)

$$\overline{\boldsymbol{X}}_t^\top \overline{\boldsymbol{X}}_t \boldsymbol{a} = \overline{\boldsymbol{X}}_t^\top (\boldsymbol{y}_t - \boldsymbol{X}_t \hat{\boldsymbol{w}}_{t-1}).$$

Therefore, we can compute $\hat{\boldsymbol{a}}_t = \overline{\boldsymbol{X}}_t^\dagger (\boldsymbol{y}_t - \boldsymbol{X}_t \hat{\boldsymbol{w}}_{t-1})$ as a global minimizer of (8), where the pseudoinverse $\overline{\boldsymbol{X}}_t^\dagger$ can be calculated via an SVD on $\overline{\boldsymbol{X}}_t$. Then a common global minimizer $\hat{\boldsymbol{w}}_t \in \mathcal{K}_t$ is given as $\hat{\boldsymbol{w}}_t = \hat{\boldsymbol{w}}_{t-1} + \boldsymbol{K}_{t-1} \hat{\boldsymbol{a}}_t$. It remains to compute an orthonormal basis matrix $\boldsymbol{K}_t$ for the intersection of the nullspaces of $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_t$, that is the intersection of the nullspace of $\boldsymbol{X}_t$ and the range space of $\boldsymbol{K}_{t-1}$. This can be done as follows. First, compute an orthonormal basis for the nullspace of $\overline{\boldsymbol{X}}_t$, then left-multiply this basis by $\boldsymbol{K}_{t-1}$. This yields the desired $\boldsymbol{K}_t$. □

Proof of Proposition 7. The proof is obtained in the same way as in Theorem 1, by exchanging the empirical part (e.g., $L_t(\boldsymbol{w}, D_t)$ and c_t) with the population part (e.g., $\mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\boldsymbol{w}; \boldsymbol{d})]$ and c_t^*). □

Proof of Theorem 2. The feasibility of the relaxed Continual Learner† follows from Proposition 7, which also suggests that

$$L_t(\overline{\boldsymbol{w}}; D_t) \leq c_t + \zeta(m_t, \delta/T), \quad \forall t = 1, \ldots, T \tag{36}$$

with probability at least $1 - \delta$. Note that in (36) the uniform convergence bound of Assumption 3 was invoked for each task, meaning that we have $\mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\overline{\boldsymbol{w}}; \boldsymbol{d})] \leq L_t(\overline{\boldsymbol{w}}; D_t) + \zeta(m_t, \delta/T)$ and $c_t \leq c_t^* + \zeta(m_t, \delta/T)$. Substitute them into (36) to get $\mathbb{E}_{\boldsymbol{d} \sim \mathcal{D}_t}[\ell_t(\overline{\boldsymbol{w}}; \boldsymbol{d})] \leq c_t^* + 3\zeta(m_t, \delta/T)$, finishing the proof. □

Proof of Theorem 3. By Proposition 7, with probability at least $1 - \delta$, we have $\overline{\boldsymbol{w}}$ satisfying

$$\frac{1}{m_T} \sum_{i=1}^{m_T} \ell_T(\overline{\boldsymbol{w}}; \boldsymbol{d}_{Ti}) + \sum_{t=1}^{T-1} \frac{1}{s_t} \sum_{i=1}^{s_t} \ell_t(\overline{\boldsymbol{w}}; \boldsymbol{d}_{ti}) \leq c_T + \zeta(m_T, \delta/T) + \sum_{t=1}^{T-1} \big(\hat{c}_t + \zeta(s_t, \delta/T)\big),$$

where $\hat{c}_t$ is the minimum of $\frac{1}{s_t}\sum_{i=1}^{s_t}\ell_t(\boldsymbol{w};\boldsymbol{d}_{ti})$ over $\mathcal{W}$, and c_T the minimum of $\frac{1}{m_T}\sum_{i=1}^{m_T}\ell_T(\boldsymbol{w};\boldsymbol{d}_{Ti})$ over $\mathcal{W}$. Then, the proof technique of Theorem 2 implies (19), and the proof is complete. $\square$